Sterling Commerce
An IBM Company

Selling and Fulfillment Foundation: Product Concepts Guide

Release 9.0

Last updated in HF 11

November 2010

Copyright Notice

STERLING COMMERCE SOFTWARE

Sterling Commerce, Inc.
4600 Lakehurst Court
Dublin, Ohio 43016-2000

Third-Party Software

Portions of the Sterling Commerce Software may include products, or may be distributed on the same storage media with products, ("Third Party Software") offered by third parties ("Third Party Licensors"). Sterling Commerce Software may include Third Party Software covered by the following copyrights: Copyright © 2006-2008 Andres Almiray. Copyright © 1999-2005 The Apache Software Foundation. Erik Arvidsson. Copyright © 2008 Azer Koçulu http://azer.kodfabrik.com. Copyright © Einar Lielmanis, einars@gmail.com. Copyright © 2006 John Reilly (www.inconspicuous.org) and Copyright © 2002 Douglas Crockford (www.crockford.com). Copyright © 2009 John Resig, http://jquery.com/. Copyright © 2006-2008 Json-lib. Copyright © 2001 LOOX Software, Inc. Copyright © 2003-2008 Luck Consulting Pty. Ltd. Copyright 2002-2004 © MetaStuff, Ltd. Copyright © 2009 Michael Mathews micmath@gmail.com. Copyright © 1999-2005 Northwoods Software Corporation. Copyright © Microsoft Corp. 1981-1998. Purple Technology, Inc. Copyright © 2004-2008 QOS.ch. Copyright © 2005 Sabre Airline Solutions. Copyright © 2004 SoftComplex, Inc. Copyright © 2000-2007 Sun Microsystems, Inc. Copyright © 2001 VisualSoft Technologies Limited. Copyright © 2001 Zero G Software, Inc. All rights reserved by all listed parties.

The Sterling Commerce Software is distributed on the same storage media as certain Third Party Software covered by the following copyrights: Copyright © 1999-2006 The Apache Software Foundation. Copyright © 2001-2003 Ant-Contrib project. Copyright © 1998-2007 Bela Ban. Copyright © 2005 Eclipse Foundation. Copyright © 2002-2006 Julian Hyde and others. Copyright © 2006-2009 Ext JS, Inc. Copyright © 1997 ICE Engineering, Inc./Timothy Gerard Endres. Copyright 2000, 2006 IBM Corporation and others. Copyright © 1987-2006 ILOG, Inc. Copyright © 2000-2006 Infragistics. Copyright © 2002-2005 JBoss, Inc. Copyright LuMriX.net GmbH, Switzerland. Copyright © 1998-2009 Mozilla.org. Copyright © 2003-2009 Mozdev Group, Inc. Copyright © 1999-2002 JBoss.org. Copyright © 2007, the OWASP Foundation. Copyright Raghu K, 2003. Copyright © 2004 David Schweinsberg. Copyright © 2005-2006 Darren L. Spurgeon. Copyright © 2005-2008 Sam Stephenson. Copyright © S.E. Morris (FISH) 2003-04. Copyright © 1998 Regents of the University of California. Copyright © 2006 VisualSoft Technologies. Copyright © 2002-2009 Zipwise Software. All rights reserved by all listed parties.

Third Party Software which is included, or are distributed on the same storage media with, the Sterling Commerce Software where use, duplication, or disclosure by the United States government or a government contractor or subcontractor, are provided with RESTRICTED RIGHTS under Title 48 CFR 2.101, 12.212, 52.227-19, 227.7201 through 227.7202-4, DFAR 252.227-7013(c) (1) (ii) and (2), DFAR 252.227-7015(b)(6/95), DFAR 227.7202-3(a), FAR 52.227-14(g)(2)(6/87), and FAR 52.227-19(c)(2) and (6/87) as applicable.

Additional information regarding certain Third Party Software is located at installdir/SCI_License.txt.

Some Third Party Licensors also provide license information and/or source code for their software via their respective links set forth below:

http://danadler.com/jacob/

http://www.dom4j.org

This product includes software developed by the Apache Software Foundation (http://www.apache.org). This product includes software developed by the Ant-Contrib project (http://sourceforge.net/projects/ant-contrib). This product includes software developed by the JDOM Project (http://www.jdom.org/). This product includes code licensed from RSA Data Security (via Sun Microsystems, Inc.). Sun, Sun Microsystems, the Sun Logo, Java, JDK, the Java Coffee Cup logo, JavaBeans, JDBC, JMX and all JMX based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. All other trademarks and logos are trademarks of their respective owners.

THE APACHE SOFTWARE FOUNDATION SOFTWARE

The Sterling Commerce Software is distributed with or on the same storage media as the following software products (or components thereof) and java source code files: Xalan version 2.5.2, Cookie.java, Header.java, HeaderElement.java, HttpException.java, HttpState.java, NameValuePair.java, CronTimeTrigger.java, DefaultTimeScheduler.java, PeriodicTimeTrigger.java, Target.java, TimeScheduledEntry.java, TimeScheduler.java, TimeTrigger.java, Trigger.java, BinaryHeap.java,

PriorityQueue.java, SynchronizedPriorityQueue.java, GetOpt.java, GetOptsException.java, IllegalArgumentException.java, MissingOptArgException.java (collectively, "Apache 1.1 Software"). Apache 1.1 Software is free software which is distributed under the terms of the following license:

License Version 1.1

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistribution in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3.The end-user documentation included with the redistribution, if any, must include the following acknowledgement: "This product includes software developed by the Apache Software Foundation (http://www.apache.org)." Alternatively, this acknowledgement may appear in the software itself, if and whenever such third-party acknowledgements normally appear.

4.The names "Commons", "Jakarta", "The Jakarta Project", "HttpClient", "log4j", "Xerces "Xalan", "Avalon", "Apache Avalon", "Avalon Cornerstone", "Avalon Framework", "Apache" and "Apache Software Foundation" must not be used to endorse or promote products derived from this software without specific prior written permission. For written permission, please contact apache@apache.org.

5.Products derived from this software may not be called "Apache", nor may "Apache" appear in their name, without the prior written permission of the Apache Software Foundation.

THIS SOFTWARE IS PROVIDED "AS IS" AND ANY EXPRESS OR IMIPLIED WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTIBILITY, AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL APACHE SOFTWARE FOUNDATION OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTIAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This software consists of voluntary contributions made by many individuals on behalf of the Apache Software Foundation. The GetOpt.java, GetOptsException.java, IlligalArgumentException.java and MissingOptArgException.java software was originally based on software copyright © 2001, Sun Microsystems, http://www.sun.com. For more information on the Apache Software Foundation, please see <http://www.apache.org/>.

The preceding license only applies to the Apache 1.1 Software and does not apply to the Sterling Commerce Software or to any other Third Party Software.

The Sterling Commerce Software is also distributed with or on the same storage media as the following software products (or components thereof): Ant, Antinstaller, Apache File Upload Package, Apache Commons Beans, Apache Commons BetWixt, Apache Commons Collection, Apache Commons Digester, Apache Commons IO, Apache Commons Lang., Apache Commons Logging, Apache Commons Net, Apache Jakarta Commons Pool, Apache Jakarta ORO, Lucene, Xerces version 2.7, Apache Log4J, Apache SOAP, Apache Struts and Apache Xalan 2.7.0, (collectively, "Apache 2.0 Software"). Apache 2.0 Software is free software which is distributed under the terms of the Apache License Version 2.0. A copy of License Version 2.0 is found in the following directory files for the individual pieces of the Apache 2.0 Software: installdir/jar/commons_upload/1_0/ CommonsFileUpload_License.txt,
installdir/jar/jetspeed/1_4/RegExp_License.txt,
installdir/ant/Ant_License.txt
<install>/jar/antInstaller/0_8/antinstaller_License.txt
<install>/jar/commons_beanutils/1_7_0/commons-beanutils.jar (/META-INF/LICENSE.txt)
<install>/jar/commons_betwixt/0_8/commons-betwixt-0.8.jar (/META-INF/LICENSE.txt)
<install>/jar/commons_collections/3_2/LICENSE.txt

<install>/jar/commons_digester/1_8/commons-digester-1.8.jar (/META-INF/LICENSE.txt)
<install>/jar/commons_io/1_4/LICENSE.txt
<install>/jar/commons_lang/2_1/Commons_Lang_License.txt
<install>/jar/commons_logging/1_0_4/commons-logging-1.0.4.jar (/META-INF/LICENSE.txt)
<install>/jar/commons_net/1_4_1/commons-net-1.4.1.jar (/META-INF/LICENSE.txt)
<install>/jar/smcfs/9.0/lucene-core-2.4.0.jar (/META-INF/LICENSE.txt)
<install>/jar/struts/2_0_11/struts2-core-2.0.11.jar (./LICENSE.txt)
<install>/jar/commons_pool/1_4/Commons_License.txt
<install>/jar/jakarta_oro/2_0_8/JakartaOro_License.txt
<install>/jar/log4j/1_2_15/LOG4J_License.txt
<install>/jar/xalan/2_7/Xalan_License.txt
<install>/jar/soap/2_3_1/Apache_SOAP_License.txt

Unless otherwise stated in a specific directory, the Apache 2.0 Software was not modified. Neither the Sterling Commerce Software, modifications, if any, to Apache 2.0 Software, nor other Third Party Code is a Derivative Work or a Contribution as defined in License Version 2.0. License Version 2.0 applies only to the Apache 2.0 Software which is the subject of the specific directory file and does not apply to the Sterling Commerce Software or to any other Third Party Software. License Version 2.0 includes the following provision:

"Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License."

NOTICE file corresponding to the section 4 d of the Apache License, Version 2.0, in this case for the Apache Ant distribution. Apache Ant Copyright 1999-2008 The Apache Software Foundation. This product includes software developed by The Apache Software Foundation (http://www.apache.org/). This product includes also software developed by:

- the W3C consortium (http://www.w3c.org)
- the SAX project (http://www.saxproject.org)

The <sync> task is based on code Copyright © 2002, Landmark Graphics Corp that has been kindly donated to the Apache Software Foundation.

Portions of this software were originally based on the following:

- software copyright © 1999, IBM Corporation., http://www.ibm.com.
- software copyright © 1999, Sun Microsystems., http://www.sun.com.
- voluntary contributions made by Paul Eng on behalf of the Apache Software Foundation that were originally developed at iClick, Inc., software copyright © 1999.

NOTICE file corresponding to the section 4 d of the Apache License, Version 2.0, in this case for the Apache Lucene distribution. Apache Lucene Copyright 2006 The Apache Software Foundation. This product includes software developed by The Apache Software Foundation (http://www.apache.org/). The snowball stemmers in contrib/snowball/src/java/net/sf/snowball were developed by Martin Porter and Richard Boulton. The full snowball package is available from http://snowball.tartarus.org/

Ant-Contrib Software

The Sterling Commerce Software is distributed with or on the same storage media as the Anti-Contrib software (Copyright © 2001-2003 Ant-Contrib project. All rights reserved.) (the "Ant-Contrib Software"). The Ant-Contrib Software is free software which is distributed under the terms of the following license:

The Apache Software License, Version 1.1

Copyright © 2001-2003 Ant-Contrib project. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1.Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2.Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. The end-user documentation included with the redistribution, if any, must include the following acknowledgement:

"This product includes software developed by the Ant-Contrib project (http://sourceforge.net/projects/ant-contrib)."

Alternately, this acknowledgement may appear in the software itself, if and wherever such third-party acknowledgements normally appear.

4. The name Ant-Contrib must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact ant-contrib-developers@lists.sourceforge.net.

5. Products derived from this software may not be called "Ant-Contrib" nor may "Ant-Contrib" appear in their names without prior written permission of the Ant-Contrib project.

THIS SOFTWARE IS PROVIDED ``AS IS'' AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE ANT-CONTRIB PROJECT OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The preceding license only applies to the Ant-Contrib Software and does not apply to the Sterling Commerce Software or to any other Third Party Software.

ANTISAMY SOFTWARE

The Sterling Commerce Software is distributed with or on the same storage media as the AntiSamy software (Copyright © 1998 Regents of the University of California. All rights reserved.) (the "AntiSamy Software"). The AntiSamy Software is free software which is distributed under the terms of the following license:

Copyright © 1998, Regents of the University of California

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

Neither the name of the <ORGANIZATION> nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,

PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

COOLBUTTONS SOFTWARE

The Sterling Commerce Software is also distributed with or on the same storage media as Coolbuttons.js ("Coolbuttons Software"), which is subject to the following license:

This Button Script was designed by Erik Arvidsson for WebFX. For more info and examples see: http://webfx.eae.net or send email to erik@eae.net. Feel free to use this code as long as this disclaimer is intact.

The preceding license only applies to the Coolbuttons Software and does not apply to the Sterling Commerce Software, or any other Third Party Software.

DOM4J Software

The Sterling Commerce Software is distributed with or on the same storage media as the Dom4h Software which is free software distributed under the terms of the following license:

Redistribution and use of this software and associated documentation ("Software"), with or without modification, are permitted provided that the following conditions are met:

1.Redistributions of source code must retain copyright statements and notices. Redistributions must also contain a copy of this document.

2.Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3.The name "DOM4J" must not be used to endorse or promote products derived from this Software without prior written permission of MetaStuff, Ltd. For written permission, please contact dom4j-info@metastuff.com.

4.Products derived from this Software may not be called "DOM4J" nor may "DOM4J" appear in their names without prior written permission of MetaStuff, Ltd. DOM4J is a registered trademark of MetaStuff, Ltd.

5.Due credit should be given to the DOM4J Project - http://www.dom4j.org

THIS SOFTWARE IS PROVIDED BY METASTUFF, LTD. AND CONTRIBUTORS ``AS IS'' AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL METASTUFF, LTD. OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The preceding license only applies to the Dom4j Software and does not apply to the Sterling Commerce Software, or any other Third Party Software.

THE ECLIPSE SOFTWARE FOUNDATION

The Sterling Commerce Software is also distributed with or on the same storage media as the following

software:

com.ibm.icu.nl1_3.4.4.v200606220026.jar, org.eclipse.ant.core.nl1_3.1.100.v200606220026.jar,
org.eclipse.ant.ui.nl1_3.2.0.v200606220026.jar, org.eclipse.compare.nl1_3.2.0.v200606220026.jar,
org.eclipse.core.boot.nl1_3.1.100.v200606220026.jar,
org.eclipse.core.commands.nl1_3.2.0.v200606220026.jar,
org.eclipse.core.contenttype.nl1_3.2.0.v200606220026.jar,
org.eclipse.core.expressions.nl1_3.2.0.v200606220026.jar,
org.eclipse.core.filebuffers.nl1_3.2.0.v200606220026.jar,
org.eclipse.core.filesystem.nl1_1.0.0.v200606220026.jar,
org.eclipse.core.jobs.nl1_3.2.0.v200606220026.jar,
org.eclipse.core.resources.nl1_3.2.0.v200606220026.jar,
org.eclipse.core.runtime.compatibility.auth.nl1_3.2.0.v200606220026.jar,
org.eclipse.core.runtime.compatibility.nl1_3.1.100.v200606220026.jar,
org.eclipse.core.runtime.nl1_3.2.0.v200606220026.jar,
org.eclipse.core.variables.nl1_3.1.100.v200606220026.jar,
org.eclipse.debug.core.nl1_3.2.0.v200606220026.jar,
org.eclipse.debug.ui.nl1_3.2.0.v200606220026.jar,
org.eclipse.equinox.common.nl1_3.2.0.v200606220026.jar,
org.eclipse.equinox.preferences.nl1_3.2.0.v200606220026.jar,
org.eclipse.equinox.registry.nl1_3.2.0.v200606220026.jar,
org.eclipse.help.appserver.nl1_3.1.100.v200606220026.jar,
org.eclipse.help.base.nl1_3.2.0.v200606220026.jar, org.eclipse.help.nl1_3.2.0.v200606220026.jar,
org.eclipse.help.ui.nl1_3.2.0.v200606220026.jar, org.eclipse.jdt.apt.core.nl1_3.2.0.v200606220026.jar,
org.eclipse.jdt.apt.ui.nl1_3.2.0.v200606220026.jar,
org.eclipse.jdt.core.manipulation.nl1_1.0.0.v200606220026.jar,
org.eclipse.jdt.core.nl1_3.2.0.v200606220026.jar,
org.eclipse.jdt.debug.ui.nl1_3.2.0.v200606220026.jar,
org.eclipse.jdt.doc.isv.nl1_3.2.0.v200606220026.jar,
org.eclipse.jdt.doc.user.nl1_3.2.0.v200606220026.jar,
org.eclipse.jdt.junit4.runtime.nl1_1.0.0.v200606220026.jar,
org.eclipse.jdt.launching.nl1_3.2.0.v200606220026.jar, org.eclipse.jdt.nl1_3.2.0.v200606220026.jar,
org.eclipse.jdt.ui.nl1_3.2.0.v200606220026.jar,
org.eclipse.jface.databinding.nl1_1.0.0.v200606220026.jar,
org.eclipse.jface.nl1_3.2.0.v200606220026.jar, org.eclipse.jface.text.nl1_3.2.0.v200606220026.jar,
org.eclipse.ltk.core.refactoring.nl1_3.2.0.v200606220026.jar,
org.eclipse.ltk.ui.refactoring.nl1_3.2.0.v200606220026.jar,
org.eclipse.osgi.nl1_3.2.0.v200606220026.jar, org.eclipse.osgi.services.nl1_3.1.100.v200606220026.jar,
org.eclipse.osgi.util.nl1_3.1.100.v200606220026.jar, org.eclipse.pde.core.nl1_3.2.0.v200606220026.jar,
org.eclipse.pde.doc.user.nl1_3.2.0.v200606220026.jar,
org.eclipse.pde.junit.runtime.nl1_3.2.0.v200606220026.jar,
org.eclipse.pde.nl1_3.2.0.v200606220026.jar, org.eclipse.pde.runtime.nl1_3.2.0.v200606220026.jar,
org.eclipse.pde.ui.nl1_3.2.0.v200606220026.jar,
org.eclipse.platform.doc.isv.nl1_3.2.0.v200606220026.jar,
org.eclipse.platform.doc.user.nl1_3.2.0.v200606220026.jar,
org.eclipse.rcp.nl1_3.2.0.v200606220026.jar, org.eclipse.search.nl1_3.2.0.v200606220026.jar,
org.eclipse.swt.nl1_3.2.0.v200606220026.jar, org.eclipse.team.core.nl1_3.2.0.v200606220026.jar,
org.eclipse.team.cvs.core.nl1_3.2.0.v200606220026.jar,
org.eclipse.team.cvs.ssh.nl1_3.2.0.v200606220026.jar,
org.eclipse.team.cvs.ssh2.nl1_3.2.0.v200606220026.jar,
org.eclipse.team.cvs.ui.nl1_3.2.0.v200606220026.jar, org.eclipse.team.ui.nl1_3.2.0.v200606220026.jar,
org.eclipse.text.nl1_3.2.0.v200606220026.jar, org.eclipse.ui.browser.nl1_3.2.0.v200606220026.jar,
org.eclipse.ui.cheatsheets.nl1_3.2.0.v200606220026.jar,
org.eclipse.ui.console.nl1_3.1.100.v200606220026.jar,
org.eclipse.ui.editors.nl1_3.2.0.v200606220026.jar,
org.eclipse.ui.externaltools.nl1_3.1.100.v200606220026.jar,
org.eclipse.ui.forms.nl1_3.2.0.v200606220026.jar, org.eclipse.ui.ide.nl1_3.2.0.v200606220026.jar,
org.eclipse.ui.intro.nl1_3.2.0.v200606220026.jar, org.eclipse.ui.navigator.nl1_3.2.0.v200606220026.jar,
org.eclipse.ui.navigator.resources.nl1_3.2.0.v200606220026.jar,
org.eclipse.ui.nl1_3.2.0.v200606220026.jar,
org.eclipse.ui.presentations.r21.nl1_3.2.0.v200606220026.jar,
org.eclipse.ui.views.nl1_3.2.0.v200606220026.jar,

org.eclipse.ui.views.properties.tabbed.nl1_3.2.0.v200606220026.jar,
org.eclipse.ui.workbench.nl1_3.2.0.v200606220026.jar,
org.eclipse.ui.workbench.texteditor.nl1_3.2.0.v200606220026.jar,
org.eclipse.update.configurator.nl1_3.2.0.v200606220026.jar,
org.eclipse.update.core.nl1_3.2.0.v200606220026.jar,
org.eclipse.update.scheduler.nl1_3.2.0.v200606220026.jar,
org.eclipse.update.ui.nl1_3.2.0.v200606220026.jar,
com.ibm.icu_3.4.4.1.jar,
org.eclipse.core.commands_3.2.0.I20060605-1400.jar,
org.eclipse.core.contenttype_3.2.0.v20060603.jar,
org.eclipse.core.expressions_3.2.0.v20060605-1400.jar,
org.eclipse.core.filesystem.linux.x86_1.0.0.v20060603.jar,
org.eclipse.core.filesystem_1.0.0.v20060603.jar, org.eclipse.core.jobs_3.2.0.v20060603.jar,
org.eclipse.core.runtime.compatibility.auth_3.2.0.v20060601.jar,
org.eclipse.core.runtime_3.2.0.v20060603.jar, org.eclipse.equinox.common_3.2.0.v20060603.jar,
org.eclipse.equinox.preferences_3.2.0.v20060601.jar, org.eclipse.equinox.registry_3.2.0.v20060601.jar,
org.eclipse.help_3.2.0.v20060602.jar, org.eclipse.jface.text_3.2.0.v20060605-1400.jar,
org.eclipse.jface_3.2.0.I20060605-1400.jar, org.eclipse.osgi_3.2.0.v20060601.jar,
org.eclipse.swt.gtk.linux.x86_3.2.0.v3232m.jar, org.eclipse.swt_3.2.0.v3232o.jar,
org.eclipse.text_3.2.0.v20060605-1400.jar,
org.eclipse.ui.workbench.texteditor_3.2.0.v20060605-1400.jar,
org.eclipse.ui.workbench_3.2.0.I20060605-1400.jar, org.eclipse.ui_3.2.0.I20060605-1400.jar,
runtime_registry_compatibility.jar, eclipse.exe, eclipse.ini, and startup.jar
(collectively, "Eclipse Software").

All Eclipse Software is distributed under the terms and conditions of the Eclipse Foundation Software User Agreement (EFSUA) and/or terms and conditions of the Eclipse Public License Version 1.0 (EPL) or other license agreements, notices or terms and conditions referenced for the individual pieces of the Eclipse Software, including without limitation "Abouts", "Feature Licenses", and "Feature Update Licenses" as defined in the EFSUA.

A copy of the Eclipse Foundation Software User Agreement is found at
<install_dir>/platformrcp/5_5/rcpdependencies/windows/eclipse/plugins/notice.html,
<install_dir>/platformrcp/5_5/rcpdependencies/windows/eclipse/plugins/notice.html,
<install_dir>/platformrcp/5_5/rcpdependencies/gtk.linux.x86/eclipse/plugins/notice.html, and
<install_dir>/platformrcp/5_5/rcpdependencies/gtk.linux.x86/eclipse/plugins/notice.html.

A copy of the EPL is found at
<install_dir>/platformrcp/5_5/rcpdependencies/windows/eclipse/plugins/epl-v10.htm,
<install_dir>/platformrcp/5_5/rcpdependencies/windows/eclipse/plugins/eclipse/epl-v10.htm,
<install_dir>/platformrcp/5_5/rcpdependencies/gtk.linux.x86/eclipse/plugins/epl-v10.html, and
<install_dir>/platformrcp/5_5/rcpdependencies/gtk.linux.x86/eclipse/plugins/epl-v10.html.

The reference to the license agreements, notices or terms and conditions governing each individual piece of the Eclipse Software is found in the directory files for the individual pieces of the Eclipse Software as described in the file identified as installdir/SCI_License.txt.

These licenses only apply to the Eclipse Software and do not apply to the Sterling Commerce Software, or any other Third Party Software.

The Language Pack (NL Pack) piece of the Eclipse Software, is distributed in object code form. Source code is available at
http://archive.eclipse.org/eclipse/downloads/drops/L-3.2_Language_Packs-200607121700/index.php. In the event the source code is no longer available from the website referenced above, contact Sterling Commerce at 978-513-6000 and ask for the Release Manager. A copy of this license is located at
<install_dir>/SI/repository/rcp/rcpdependencies/windows/eclipse/plugins/epl-v10.htm and
<install_dir>/SI/repository/rcp/rcpdependencies/gtk.linux.x86/eclipse/plugins/epl-v10.html.

The org.eclipse.core.runtime_3.2.0.v20060603.jar piece of the Eclipse Software was modified slightly in order to remove classes containing encryption items. The org.eclipse.core.runtime_3.2.0.v20060603.jar was modified to remove the Cipher, CipherInputStream and CipherOutputStream classes and rebuild the org.eclipse.core.runtime_3.2.0.v20060603.jar.

Ehcache Software

The Sterling Commerce Software is also distributed with or on the same storage media as the Ehcache v.1.5 software (Copyright © 2003-2008 Luck Consulting Pty. Ltd.) ("Ehcache Software"). Ehcache Software is free software which is distributed under the terms of the Apache License Version 2.0. A copy of License Version 2.0 is found in <install>/jar/smcfs/9.0/ehcache-1.5.0.jar (./LICENSE.txt).

The Ehcache Software was not modified. Neither the Sterling Commerce Software, modifications, if any, to the Ehcache Software, nor other Third Party Code is a Derivative Work or a Contribution as defined in License Version 2.0. License Version 2.0 applies only to the Ehcache Software which is the subject of the specific directory file and does not apply to the Sterling Commerce Software or to any other Third Party Software. License Version 2.0 includes the following provision:

"Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License."

ESAPI SOFTWARE

The Sterling Commerce Software is also distributed with or on the same storage media

as the ESAPI software (Copyright © 2007, the OWASP Foundation) ("ESAPI Software"). ESAPI Software Software is free software which is distributed under the terms of the following license:

Copyright © 2007, The OWASP Foundation

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

Neither the name of the OWASP Foundation nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

EZMorph Software

The Sterling Commerce Software is also distributed with or on the same storage media as the EZMorph v. 1.0.4 software (Copyright © 2006-2008 Andres Almiray) ("EZMorph Software"). EZMorph Software is free software which is distributed under the terms of the Apache License Version 2.0. A copy of License

Version 2.0 is found in <install>/jar/ezmorph/1_0_4/ezmorph-1.0.4.jar (./LICENSE.txt).

The EZMorph Software was not modified. Neither the Sterling Commerce Software, modifications, if any, to the EZMorph Software, nor other Third Party Code is a Derivative Work or a Contribution as defined in License Version 2.0. License Version 2.0 applies only to the EZMorph Software which is the subject of the specific directory file and does not apply to the Sterling Commerce Software or to any other Third Party Software. License Version 2.0 includes the following provision:

"Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License."

Firebug Lite Software

The Sterling Commerce Software is distributed with or on the same storage media as the Firebug Lite Software which is free software distributed under the terms of the following license:

Copyright © 2008 Azer Koçulu http://azer.kodfabrik.com. All rights reserved.

Redistribution and use of this software in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

* Neither the name of Azer Koçulu. nor the names of any other contributors may be used to endorse or promote products derived from this software without specific prior written permission of Parakey Inc.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

JAVASCRIPT MINIFIER

The Sterling Commerce Software is distributed with or on the same storage media as the JSMin Software which is free software distributed under the terms of the following license:

JSMin.java 2006-02-13; Updated 2007-08-20 with updates from jsmin.c (2007-05-22)

Copyright © 2006 John Reilly (www.inconspicuous.org)

This work is a translation from C to Java of jsmin.c published by Douglas Crockford. Permission is hereby granted to use the Java version under the same conditions as the jsmin.c on which it is based.

jsmin.c 2003-04-21

Copyright © 2002 Douglas Crockford (www.crockford.com)

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including

without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

The Software shall be used for Good, not Evil.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

ICE SOFTWARE

The Sterling Commerce Software is distributed on the same storage media as the ICE Software (Copyright © 1997 ICE Engineering, Inc./Timothy Gerard Endres.) ("ICE Software"). The ICE Software is independent from and not linked or compiled with the Sterling Commerce Software. The ICE Software is a free software product which can be distributed and/or modified under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License or any later version.

A copy of the GNU General Public License is provided at installdir/jar/jniregistry/1_2/ICE_License.txt. This license only applies to the ICE Software and does not apply to the Sterling Commerce Software, or any other Third Party Software.

The ICE Software was modified slightly in order to fix a problem discovered by Sterling Commerce involving the RegistryKey class in the RegistryKey.java in the JNIRegistry.jar. The class was modified to comment out the finalize () method and rebuild of the JNIRegistry.jar file.

Source code for the bug fix completed by Sterling Commerce on January 8, 2003 is located at: installdir/jar/jniregistry/1_2/RegistryKey.java. Source code for all other components of the ICE Software is located at http://www.trustice.com/java/jnireg/index.shtml.

The ICE Software is distributed WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

JBOSS SOFTWARE

The Sterling Commerce Software is distributed on the same storage media as the JBoss Software (Copyright © 1999-2002 JBoss.org) ("JBoss Software"). The JBoss Software is independent from and not linked or compiled with the Sterling Commerce Software. The JBoss Software is a free software product which can be distributed and/or modified under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License or any later version.

A copy of the GNU Lesser General Public License is provided at:
<install_dir>\jar\jboss\4_2_0\LICENSE.html

This license only applies to the JBoss Software and does not apply to the Sterling Commerce Software, or any other Third Party Software.

The JBoss Software is not distributed by Sterling Commerce in its entirety. Rather, the distribution is limited to the following jar files: el-api.jar, jasper-compiler-5.5.15.jar, jasper-el.jar, jasper.jar, jboss-common-client.jar, jboss-j2ee.jar, jboss-jmx.jar, jboss-jsr77-client.jar, jbossmq-client.jar, jnpserver.jar, jsp-api.jar, servlet-api.jar, tomcat-juli.jar.

The JBoss Software was modified slightly in order to allow the ClientSocketFactory to return a socket connected to a particular host in order to control the host interfaces, regardless of whether the

ClientSocket Factory specified was custom or note. Changes were made to org.jnp.server.Main. Details concerning this change can be found at http://sourceforge.net/tracker/?func=detail&aid=1008902&group_id=22866&atid=376687.

Source code for the modifications completed by Sterling Commerce on August 13, 2004 is located at: http://sourceforge.net/tracker/?func=detail&aid=1008902&group_id=22866&atid=376687. Source code for all other components of the JBoss Software is located at http://www.jboss.org.

JGO SOFTWARE

The Sterling Commerce Software is distributed with, or on the same storage media, as certain redistributable portions of the JGo Software provided by Northwoods Software Corporation under a commercial license agreement (the "JGo Software"). The JGo Software is provided subject to the disclaimers set forth above and the following notice:

U.S. Government Restricted Rights

The JGo Software and documentation are provided with RESTRICTED RIGHTS. Use, duplication, or disclosure by the Government is subject to restrictions as set forth in subparagraph (C)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013 or subparagraphs (C)(1) and (2) of the Commercial Computer Software - Restricted Rights at 48 CFR 52.227-19, as applicable. Contractor / manufacturer of the JGo Software is Northwoods Software Corporation, 142 Main St., Nashua, NH 03060.

JSDoc Tookit Software

The Sterling Commerce Software is distributed with or on the same storage media as the JSDoc Toolkit software (Copyright © 2008 Michael Mathews) ("JSDoc Toolkit Software"), which is subject to the following license:

All code specific to JsDoc Toolkit are free, open source and licensed for use under the X11/MIT License.

JsDoc Toolkit is Copyright © 2008 Michael Mathews <micmath@gmail.com>

This program is free software; you can redistribute it and/or modify it under the terms below.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions: The above copyright notice and this permission notice must be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

JSLib Software

The Sterling Commerce Software is distributed with or on the same storage media as the JSLib software product (Copyright © 2003-2009 Mozdev Group, Inc.) ("JSLib Software"). The JSLib Software is distributed under the terms of the MOZILLA PUBLIC LICENSE Version 1.1. A copy of this license is located at <install>/repository/eardata/platform_uifwk_ide/war/designer/MPL-1.1.txt. The JSLib Software code is distributed in source form and is located at http://jslib.mozdev.org/installation.html. Neither the Sterling Commerce Software nor any other Third Party Code is a Modification or Contribution subject to the Mozilla Public License. Pursuant to the terms of the Mozilla Public License, the following

notice applies only to the JSLib Software (and not to the Sterling Commerce Software or any other Third Party Software):

"The contents of the file located at http://www.mozdev.org/source/browse/jslib/ are subject to the Mozilla Public License Version 1.1 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.mozilla.org/MPL/

Software distributed under the License is distributed on an "AS IS" basis, WITHOUT WARRANTY OF ANY KIND, either express or implied. See the License for the specific language governing rights and limitations under the License.

The Original Code is Mozdev Group, Inc. code. The Initial Developer of the Original Code is Mozdev Group, Inc. Portions created by_Mozdev Group, Inc. are Copyright © 2003 Mozdev Group, Inc. All Rights Reserved. Original Author: Pete Collins <pete@mozdev.org>one Contributor(s):_____none listed_____.

Alternatively, the contents of this file may be used under the terms of the ____ license (the "[___] License"), in which case the provisions of [___] License are applicable instead of those above. If you wish to allow use of your version of this file only under the terms of the [___] License and not allow others to use your version of this file under the MPL, indicate your decision by deleting the provisions above and replace them with the notice and other provisions required by the [___] License. If you do not delete the provisions above, a recipient may use your version of this file under either the MPL or the [___] License."

The preceding license only applies to the JSLib Software and does not apply to the Sterling Commerce Software, or any other Third Party Software.

Json Software

The Sterling Commerce Software is also distributed with or on the same storage media as the Json 2.2.2 software (Copyright © 2006-2008 Json-lib) ("Json Software"). Json Software is free software which is distributed under the terms of the Apache License Version 2.0. A copy of License Version 2.0 is found in <install>/jar/jsonlib/2_2_2/json-lib-2.2.2-jdk13.jar.

This product includes software developed by Douglas Crockford (http://www.crockford.com).

The Json Software was not modified. Neither the Sterling Commerce Software, modifications, if any, to the Json Software, nor other Third Party Code is a Derivative Work or a Contribution as defined in License Version 2.0. License Version 2.0 applies only to the Json Software which is the subject of the specific directory file and does not apply to the Sterling Commerce Software or to any other Third Party Software. License Version 2.0 includes the following provision:

"Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License."

Prototype Software

The Sterling Commerce Software is distributed with or on the same storage media as the Prototype software (Copyright © 2005-2008 Sam Stephenson) ("Prototype Software"), which is subject to the following license:

Copyright © 2005-2008 Sam Stephenson

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the

following conditions:

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Purple Technology

The Sterling Commerce Software is distributed with or on the same storage media as the Purple Technology Software (Copyright © 1995-1999 Purple Technology, Inc.) ("Purple Technology Software"), which is subject to the following license:

Copyright © 1995-1999 Purple Technology, Inc. All rights reserved.

PLAIN LANGUAGE LICENSE: Do whatever you like with this code, free of charge, just give credit where credit is due. If you improve it, please send your improvements to alex@purpletech.com. Check http://www.purpletech.com/code/ for the latest version and news.

LEGAL LANGUAGE LICENSE: Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. The names of the authors and the names "Purple Technology," "Purple Server" and "Purple Chat" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact server@purpletech.com.

THIS SOFTWARE IS PROVIDED BY THE AUTHORS AND PURPLE TECHNOLOGY "AS IS'' AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHORS OR PURPLE TECHNOLOGY BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The preceding license only applies to the Purple Technology Software and does not apply to the Sterling Commerce Software, or any other Third Party Software.

Rico Software

The Sterling Commerce Software is also distributed with or on the same storage media as the Rico.js software (Copyright © 2005 Sabre Airline Solutions) ("Rico Software"). Rico Software is free software which is distributed under the terms of the Apache License Version 2.0. A copy of License Version 2.0 is found in <install>/repository/eardata/platform/war/ajax/scripts/Rico_License.txt.

The Rico Software was not modified. Neither the Sterling Commerce Software, modifications, if any, to the Rico Software, nor other Third Party Code is a Derivative Work or a Contribution as defined in License Version 2.0. License Version 2.0 applies only to the Rico Software which is the subject of the specific directory file and does not apply to the Sterling Commerce Software or to any other Third Party Software. License Version 2.0 includes the following provision:

"Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each

Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License."

Rhino Software

The Sterling Commerce Software is distributed with or on the same storage media as the Rhino js.jar (Copyright © 1998-2009 Mozilla.org.) ("Rhino Software"). A majority of the source code for the Rhino Software is dual licensed under the terms of the MOZILLA PUBLIC LICENSE Version 1.1. or the GPL v. 2.0. Additionally, some files (at a minimum the contents of toolsrc/org/Mozilla/javascript/toolsdebugger/treetable) are available under another license as set forth in the directory file for the Rhino Software.

Sterling Commerce's use and distribution of the Rhino Software is under the Mozilla Public License. A copy of this license is located at <install>/jar/rhino/1_7R1/License.txt. The Rhino Software code is distributed in source form and is located at http://mxr.mozilla.org/mozilla/source/js/rhino/src/. Neither the Sterling Commerce Software nor any other Third Party Code is a Modification or Contribution subject to the Mozilla Public License. Pursuant to the terms of the Mozilla Public License, the following notice applies only to the Rhino Software (and not to the Sterling Commerce Software or any other Third Party Software):

"The contents of the file located at <install>/jar/rhino/1_7R1/js.jar are subject to the Mozilla Public License Version 1.1 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.mozilla.org/MPL/

Software distributed under the License is distributed on an "AS IS" basis, WITHOUT WARRANTY OF ANY KIND, either express or implied. See the License for the specific language governing rights and limitations under the License.

The Original Code is Rhino code, released May 6, 1999. The Initial Developer is Netscape Communications Corporation. Portions created by the Initial Developer are Copyright © 1997-1999. All Rights Reserved. Contributor(s):_____none listed.

The preceding license only applies to the Rico Software and does not apply to the Sterling Commerce Software, or any other Third Party Software

SLF4J Software

The Sterling Commerce Software is also distributed with or on the same storage media as the SLF4J software (Copyright © 2004-2008) ("SLF4J Software"), which is subject to the following license:

Copyright © 2004-2008 QOS.ch All rights reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Sun Microsystems

The Sterling Commerce Software is distributed with or on the same storage media

as the following software products (or components thereof): Sun JMX, and Sun JavaMail (collectively, "Sun Software"). Sun Software is free software which is distributed under the terms of the licenses issued by Sun which are included in the directory files located at:

SUN COMM JAR -installdir/jar/comm/2_0
SUN ACTIVATION JAR -installdir/jar/jaf/1_0_2
SUN JavaMail -installdir/jar/javamail/1_4

The Sterling Commerce Software is also distributed with or on the same storage media as the Web-app_2_3.dtd software (Copyright © 2007 Sun Microsystems, Inc.) ("Web-App Software"). Web-App Software is free software which is distributed under the terms of the Common Development and Distribution License ("CDDL"). A copy of <install>/repository/eardata/platform/war/WEB-INF/web_app_License.txt.

The source code for the Web-App Software may be found at:http://java.sun.com/dtd/.

Such licenses only apply to the Sun product which is the subject of such directory and does not apply to the Sterling Commerce Software or to any other Third Party Software.

The Sterling Commerce Software is also distributed with or on the same storage media as the Sun Microsystems, Inc. Java (TM) look and feel Graphics Repository ("Sun Graphics Artwork"), subject to the following terms and conditions:

Copyright 2000 by Sun Microsystems, Inc. All Rights Reserved.

Sun grants you ("Licensee") a non-exclusive, royalty free, license to use, and redistribute this software graphics artwork, as individual graphics or as a collection, as part of software code or programs that you develop, provided that i) this copyright notice and license accompany the software graphics artwork; and ii) you do not utilize the software graphics artwork in a manner which is disparaging to Sun. Unless enforcement is prohibited by applicable law, you may not modify the graphics, and must use them true to color and unmodified in every way.

This software graphics artwork is provided "AS IS," without a warranty of any kind. ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE HEREBY EXCLUDED. SUN AND ITS LICENSORS SHALL NOT BE LIABLE FOR ANY DAMAGES SUFFERED BY LICENSEE AS A RESULT OF USING, MODIFYING OR DISTRIBUTING THE SOFTWARE GRAPHICS ARTWORK.

IN NO EVENT WILL SUN OR ITS LICENSORS BE LIABLE FOR ANY LOST REVENUE, PROFIT OR DATA, OR FOR DIRECT, INDIRECT, SPECIAL, CONSEQUENTIAL, INCIDENTAL OR PUNITIVE DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF THE USE OF OR INABILITY TO USE SOFTWARE GRAPHICS ARTWORK, EVEN IF SUN HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

If any of the above provisions are held to be in violation of applicable law, void, or unenforceable in any jurisdiction, then such provisions are waived to the extent necessary for this Disclaimer to be otherwise enforceable in such jurisdiction.

The preceding license only applies to the Sun Graphics Artwork and does not apply to the Sterling Commerce Software, or any other Third Party Software.

WARRANTY DISCLAIMER

This documentation and the Sterling Commerce Software which it describes are licensed either "AS IS" or with a limited warranty, as set forth in the Sterling Commerce license agreement. Other than any limited warranties provided, NO OTHER WARRANTY IS EXPRESSED AND NONE SHALL BE IMPLIED,

INCLUDING THE WARRANTIES OF MERCHANTABILITY AND FITNESS FOR USE OR FOR A PARTICULAR PURPOSE. The applicable Sterling Commerce entity reserves the right to revise this publication from time to time and to make changes in the content hereof without the obligation to notify any person or entity of such revisions or changes.

The Third Party Software is provided "AS IS" WITHOUT ANY WARRANTY AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. FURTHER, IF YOU ARE LOCATED OR ACCESSING THIS SOFTWARE IN THE UNITED STATES, ANY EXPRESS OR IMPLIED WARRANTY REGARDING TITLE OR NON-INFRINGEMENT ARE DISCLAIMED.

Without limiting the foregoing, the ICE Software and JBoss Software are distributed WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

Contents

2 Application Platform Technology Concepts

3 Participant Management

10 Value-Added Services

13 Logistics Management

Preface

This manual provides general concepts and explanations about different features within the Application Console.

Intended Audience

This manual is intended to provide a high-level overview to all users of the Application Console.

Structure

This manual contains the following sections:

Chapter 1, "Overview of Selling and Fulfillment Foundation"

This chapter introduces the Selling and Fulfillment Foundation suite of applications and details how it can be used in different business environments.

Chapter 2, "Application Platform Technology Concepts"

This chapter introduces the technological infrastructure of the Application Platform.

Chapter 3, "Participant Management"

This chapter details Participant Management and its function in setting up organizations.

Chapter 4, "Process Modeling Concepts"

This chapter explains how Process Modeling is used to set up process-type pipelines.

Chapter 5, "Catalog Management"

This chapter describes the Catalog Management system and its components.

Chapter 6, "Global Inventory Visibility"

This chapter describes different inventory concepts that must be taken into consideration when setting up Selling and Fulfillment Foundation.

Chapter 7, "Order Management"

This chapter explains how Order Management uses the process-type pipelines developed in Process Modeling to send an order document through its various stages and track its lifecycle.

Chapter 8, "Opportunity Management"

This chapter provides a brief description of an opportunity and quotes. For additional information about opportunity management, refer to *Selling and Fulfillment Foundation: Setting Up Quotes*.

Chapter 9, "Order Promising and Scheduling"

This chapter describes how scheduling occurs for orders, based on location, inventory, and shipping options and other conditions.

Chapter 10, "Value-Added Services"

This chapter explains Value Added Services, which are used to fulfil buyer special requirements, using Work Orders.

Chapter 11, "Supply Collaboration"

This chapter describes how Supply Collaboration manages a purchase order process for your products in Selling and Fulfillment Foundation .

Chapter 12, "Payment Systems"

This chapter describes how Selling and Fulfillment Foundation carries out the critical payment related processes during order management

processing and enables you to integrate with external payment processing systems.

Chapter 13, "Logistics Management"

This chapter describes planning and monitoring order line level delivery processes so that an order is shipped when and how a customer wants it.

Chapter 14, "Reverse Logistics"

This chapter describes how Reverse Logistics manages the return process for your products.

Selling and Fulfillment Foundation Documentation

For more information about Selling and Fulfillment Foundation components, see the following manuals:

- *Selling and Fulfillment Foundation: Release Notes*
- *Selling and Fulfillment Foundation: Installation Guide*
- *Selling and Fulfillment Foundation: Upgrade Guide*
- *Selling and Fulfillment Foundation: Configuration Deployment Tool Guide*
- *Selling and Fulfillment Foundation: Performance Management Guide*
- *Selling and Fulfillment Foundation: High Availability Guide*
- *Selling and Fulfillment Foundation: System Management Guide*
- *Selling and Fulfillment Foundation: Localization Guide*
- *Selling and Fulfillment Foundation: Customization Basics Guide*
- *Selling and Fulfillment Foundation: Customizing APIs Guide*
- *Selling and Fulfillment Foundation: Customizing Console JSP Interface for End User Guide*
- *Selling and Fulfillment Foundation: Customizing the RCP Interface Guide*
- *Selling and Fulfillment Foundation: Customizing User Interfaces for Mobile Devices Guide*

- *Selling and Fulfillment Foundation: Customizing Web UI Framework Guide*
- *Selling and Fulfillment Foundation: Customizing Swing Interface Guide*
- *Selling and Fulfillment Foundation: Extending the Condition Builder Guide*
- *Selling and Fulfillment Foundation: Extending the Database Guide*
- *Selling and Fulfillment Foundation: Extending Transactions Guide*
- *Selling and Fulfillment Foundation: Using Sterling RCP Extensibility Tool Guide*
- *Selling and Fulfillment Foundation: Integration Guide*
- *Selling and Fulfillment Foundation: Product Concepts Guide*
- *Sterling Warehouse Management™ System: Concepts Guide*
- *Selling and Fulfillment Foundation: Application Platform Configuration Guide*
- *Sterling Distributed Order Management™: Configuration Guide*
- *Sterling Supply Collaboration: Configuration Guide*
- *Sterling Global Inventory Visibility™: Configuration Guide*
- *Catalog Management™: Configuration Guide*
- *Sterling Logistics Management: Configuration Guide*
- *Sterling Reverse Logistics™: Configuration Guide*
- *Sterling Warehouse Management System: Configuration Guide*
- *Selling and Fulfillment Foundation: Application Platform User Guide*
- *Sterling Distributed Order Management: User Guide*
- *Sterling Supply Collaboration: User Guide*
- *Sterling Global Inventory Visibility: User Guide*
- *Sterling Logistics Management: User Guide*
- *Sterling Reverse Logistics: User Guide*
- *Sterling Warehouse Management System: User Guide*

- *Selling and Fulfillment Foundation: Mobile Application User Guide*
- *Selling and Fulfillment Foundation: Business Intelligence Guide*
- *Selling and Fulfillment Foundation: Javadocs*
- *Sterling Selling and Fulfillment Suite™: Glossary*
- *Parcel Carrier: Adapter Guide*
- *Visual Modeler™: Application Guide*
- *Selling and Fulfillment Foundation: Multitenant Enterprise Guide*
- *Selling and Fulfillment Foundation: Password Policy Management Guide*
- *Selling and Fulfillment Foundation: Properties Guide*
- *Catalog Management: Concepts Guide*
- *Selling and Fulfillment Foundation: Pricing Concepts Guide*
- *Selling and Fulfillment Foundation: Setting Up Quotes*
- *Sterling Sensitive Data Capture Server, Release 1.0: Configuration Guide*
- *Sterling Sensitive Data Capture Server, Release 1.0: PA-DSS Implementation Guide*
- *Selling and Fulfillment Foundation: Secure Deployment Guide*
- *Business Center: Item Administration Guide*
- *Business Center: Pricing Administration Guide*
- *Business Center: Customization Guide*
- *Business Center: Localization Guide*

Conventions

The following conventions may be used in this manual:

Convention	Meaning
. . .	Ellipsis represents information that has been omitted.
< >	Angle brackets indicate user-supplied input.

Convention	Meaning
`mono-spaced text`	Mono-spaced text indicates a file name, directory path, attribute name, or an inline code example or command.
/ or \	Slashes and backslashes are file separators for Windows, UNIX, and Linux operating systems. The file separator for the Windows operating system is "\" and the file separator for UNIX and Linux systems is "/". The UNIX convention is used unless otherwise mentioned.
<INSTALL_DIR>	User-supplied location of the Selling and Fulfillment Foundation installation directory. This is only applicable for Release 8.0 and later.
<INSTALL_DIR_OLD>	User-supplied location of the Selling and Fulfillment Foundation installation directory (for Release 8.0 and later). **Note:** This is applicable only for users upgrading from Release 8.0 and later.
<SSDCS_DIR>	User-supplied location of the Sterling Sensitive Data Capture Server installation directory. This is applicable for Selling and Fulfillment Foundation, Release 9.0 and later.
<YANTRA_HOME>	User-supplied location of the Sterling Supply Chain Applications installation directory. This is only applicable for Releases 7.7, 7.9, and 7.11.
<YANTRA_HOME_OLD>	User-supplied location of the Sterling Supply Chain Applications installation directory (for Releases 7.7, 7.9, or 7.11). **Note:** This is applicable only for users upgrading from Releases 7.7, 7.9, or 7.11.
<YFS_HOME>	For Releases 7.3, 7.5, and 7.5 SP1, this is the user-supplied location of the Sterling Supply Chain Applications installation directory. For Releases 7.7, 7.9, and 7.11, this is the user-supplied location of the <YANTRA_HOME>/`Runtime` directory. For Release 8.0 and later, the <YANTRA_HOME>/`Runtime` directory is no longer used and has been substituted with the location <INSTALL_DIR>.

Convention	Meaning
<YFS_HOME_OLD>	This is the <YANTRA_HOME>/Runtime directory for Releases 7.7, 7.9, or 7.11. **Note:** This is only applicable for users upgrading from Releases 7.7, 7.9, or 7.11.
<ANALYTICS_HOME>	User-supplied location of the Sterling Analytics installation directory. **Note:** This convention is used only in the *Selling and Fulfillment Foundation: Business Intelligence Guide*.
<COGNOS_HOME>	User-supplied location of the IBM Cognos 8 Business Intelligence installation directory. **Note:** This convention is used only in the *Selling and Fulfillment Foundation: Business Intelligence Guide*.
<MQ_JAVA_INSTALL_PATH>	User-supplied location of the IBM WebSphere® MQ Java components installation directory. **Note:** This convention is used only in the *Selling and Fulfillment Foundation: System Management and Administration Guide*.
<DB>	Refers to Oracle®, IBM DB2®, or Microsoft SQL Server® depending on the database server.
<DB_TYPE>	Depending on the database used, considers the value oracle, db2, or sqlserver.

Note: The Selling and Fulfillment Foundation documentation set uses the following conventions in the context of the product name:

- Yantra is used for Release 7.7 and earlier.
- Sterling Supply Chain Applications is used for Releases 7.9 and 7.11.
- Sterling Multi-Channel Fulfillment Solution is used for Releases 8.0 and 8.2.
- Selling and Fulfillment Foundation is used for Releases 8.5 and 9.0.

1

Overview of Selling and Fulfillment Foundation

Businesses managing their extended supply chains using ERP or legacy applications face problems when trying to collaborate with business partners. Often, their applications were developed to automate internal business processes, but were not built on a platform designed to manage operations in this "extended" environment. As a result, they use costly manual processes or develop custom applications in an effort to manage their extended supply chains. Up to 60% of operational costs can be impacted by decisions outside the four walls of a company.

Selling and Fulfillment Foundation leverages the capabilities of existing enterprise applications to more effectively control this complex environment. Selling and Fulfillment Foundation enables you to extend the capabilities of existing systems or sit across multiple internal applications to provide consolidated visibility and control of key supply chain processes. Selling and Fulfillment Foundation can also be used as your primary supply chain management system.

This chapter describes the key components of Selling and Fulfillment Foundation.

1.1 The Application Platform

The Application Platform defines the technical infrastructure of Selling and Fulfillment Foundation. It contains your base level customization and enables Selling and Fulfillment Foundation to interoperate with other

systems. Refer to the following chapters for more information about the Application Platform:

- Chapter 2, "Application Platform Technology Concepts" introduces technical concepts pertaining to the infrastructure
- Chapter 3, "Participant Management" introduces business concepts pertaining to how you define your business partners
- Chapter 4, "Process Modeling Concepts" introduces business concepts pertaining to how you define your work flow

1.2 Application Console

The Application Console layer provides visibility to information using XML/EDI, UI, and wireless presentation, and through the creation of portals. This enables highly effective management to be obtained through the ability to integrate with the applications that are present in the majority of enterprise supply chains.

1.3 Selling and Fulfillment Foundation

Selling and Fulfillment Foundation is grouped into product suites that address specific, extended supply chain business problems. Customers may use some or all of the suite applications depending on their unique needs.

Selling and Fulfillment Foundation is defined by a set of APIs, business processes, and user interfaces (or consoles) that address a specific business problem.

The Selling and Fulfillment Foundation layer consists of the following applications and optional components:

Applications:

- Distributed Order Management
- Supply Collaboration
- Global Inventory Visibility
- Logistics Management
- Reverse Logistics

- Catalog Management
- Sterling Warehouse Management System

1.3.1 Distributed Order Management

The Selling and Fulfillment Foundation Distributed Order Management application provides highly configurable order management capabilities for all types of customer orders (products and services). It aggregates, manages and monitors orders from all channels, and coordinates fulfillment processes across the extended enterprise. Distributed Order Management checks for inventory availability and provides rule-based, dynamic allocation across all internal and external fulfillment locations. Moreover, it coordinates critical third-party services, such as credit, logistics and installation, and enables collaborative execution among all involved participants. It provides a single order repository and allows customers, channels, suppliers, and trading partners access to real-time order information throughout the entire fulfillment lifecycle. Distributed Order Management delivers complete flexibility for handling multiple order fulfillment processes in a single instance, and handles dynamic variations in order processes with event-driven and rule-based order coordination.

1.3.2 Supply Collaboration

The Sterling Supply Collaboration application enables the aggregation, routing and tracking of planned orders and purchase orders in an extended enterprise environment with multiple divisions and complex supplier networks. The Supply Collaboration application incorporates varying business rules by division, supplier, partners, etc. that impact how purchase orders are to be allocated, tracked and managed. It enables role-based visibility to all purchase order information, and acts as a central purchase order repository for purchase orders generated from multiple different purchasing systems or locations, even if they are external. It can be tightly linked to the Global Inventory Visibility application to provide visibility to expected inventory that is inbound, on purchase, or planned (scheduled).

1.3.3 Global Inventory Visibility

The Sterling Global Inventory Visibilityapplication coordinates global inventory across multiple sites, enterprises and participants. Managers

can track inventory at internal and external ship nodes. It provides a real-time availability picture by synchronizing multiple demand (schedules, plans, quotes, orders, etc.) and supply types (on-hand, inbound, on purchase, scheduled, etc.). It identifies shortages and allows inventory planners to resolve problems by manipulating inventory balances, through allocation of sales orders, execution of purchases or movement of inventory. Data can be shared with external systems, customers, suppliers, and partners for demand and supply management. It provides the global visibility, ATP/ATD, reservations and tracking for an extended enterprise environment.

1.3.4 Logistics Management

The Sterling Logistics Management application provides the capabilities for managing and executing an inbound or outbound delivery process. It accepts, stores and then manages the execution of a delivery plan accounting for complex, multi-step, multi-leg, and multi-mode movement of goods, including practices such as merge-in-transit, continuous movement, lane optimization and cross-docking. It coordinates all activities among all parties in the delivery chain, and proactively monitors events and notifies participants when deviations have occurred. Shipment and delivery records are tied to the original sales or purchase orders for management of dependencies among orders and shipments. It provides post-delivery reconciliation of performance, comparing actual vs. promised, SLA metric analysis, participant performance, and so forth.

1.3.5 Reverse Logistics

The Sterling Reverse Logistics application delivers condition-based returns processing, including execution and management of associated processes, such as exchange orders, refurbishment and repair requests, and return disposition. With chained order capability, Reverse Logistics can link multiple returns or repair requests to original sales orders, providing repair lifecycle tracking. It closes the returns loop, managing reverse inventory tracking back to the appropriate node based upon appropriate business rules. It handles return receipts, disposition, and initiates the crediting process.

1.3.6 Catalog Management

The Catalog Management aggregates and manages detailed product and catalog data across multiple divisions, enterprises and participants. It acts as a multi-tenant management tool that supports sharing and collaboration. It enables categorization, product cross-sell, up-sell, substitution and other features.

1.3.7 Sterling Warehouse Management System

The Selling and Fulfillment Foundation Warehouse Management application is a scalable solution for high-volume pick, pack, personalize and ship operations. Designed to make distribution more efficient and cost-effective, it includes transportation, yard and world-class logistics management. Key capabilities include planning, execution and measurement tools, and business rule-based operations.

For more information about the Sterling Warehouse Management System, see the *Sterling Warehouse Management System: Concepts Guide*.

1.4 Selling and Fulfillment Foundation Product Suites

The Sterling Warehouse Management System is grouped into product suites that address specific, extended supply chain business problems. Customers may use some or all of the suite applications depending on their unique needs.

Sterling Commerce sells and licenses its applications that are deployed as a product suite to a targeted business problem or area because businesses infrequently require only one product as they use Selling and Fulfillment Foundation for managing some of their most critical processes: order-to-cash, purchase-to-pay, and after-market service management. Therefore, Sterling Commerce markets and offers solution areas or suites of products that together deliver the primary functionality to address the particular challenges of these processes in today's networked business environment.

- Customer Fulfillment
- Supply Collaboration

- Service Parts Logistics
- Supply Chain Event Management

1.4.1 Customer Fulfillment

The Selling and Fulfillment Foundation Customer Fulfillment product suite helps companies coordinate all the supply chain processes associated with fulfilling a customer order. Companies with multiple sales channels, complex fulfillment processes or third-party fulfillment models find these applications particularly powerful.

Customer Fulfillment focuses on the requirements of managing and fulfilling customer orders in a complex, extended enterprise environment. The suite enables companies to present a single face to customers across multiple channels or interaction points, while making the complex supply chain transparent. It coordinates all fulfillment activities throughout the customer order lifecycle, such as order capture, promising, modification, allocation, fulfillment, and delivery. The Selling and Fulfillment Foundation Customer Fulfillment product suite includes the following products:

- Application Platform
- Supply Chain Event Management
- Distributed Order Management
- Global Inventory Visibility
- Reverse Logistics
- Logistics Management

1.4.2 Supply Collaboration

The Sterling Supply Collaboration product suite enables companies to coordinate inbound supply processes across multiple internal business units and divisions. It also helps companies better control replenishment processes for distribution locations they do not own or control.

Supply Collaboration focuses on the requirements for coordinating and collaborating on direct material supply plans, purchase orders, replenishment, inventory, and inbound fulfillment across multiple business units, division, suppliers, outsourced manufacturing, and

transportation providers. The suite provides the central point for collaborating on supply requirements across internal systems or divisions, sourcing orders across suppliers, and providing global visibility into inventory and inbound fulfillment. The Sterling Supply Collaboration product suite includes the following products:

- Application Platform
- Supply Chain Event Management
- Supply Collaboration
- Logistics Management
- Global Inventory Visibility

1.4.3 Service Parts Logistics

The Selling and Fulfillment Foundation Service Logistics product suite enables companies to streamline logistics processes associated with maximizing client profitability and retention beyond the initial product sale. They provide critical execution and event-driven exception management capabilities across a complex service parts network.

The Selling and Fulfillment Foundation Service Parts Logistics product suite focuses on aftermarket service operations, including returns, spares management and maintenance/ service order management. It leverages Global Inventory Visibility for managing and maintaining inventory visibility across a complex network of service centers, forward stocking locations, dealers, and so forth. It further focuses on the coordination of third-party delivered services, and the execution of the complete reverse logistics process, including return, exchange, refurbishment and disposition. The Selling and Fulfillment Foundation Service Parts Logistics product suite includes the following products:

- Application Platform
- Supply Chain Event Management
- Distributed Order Management
- Global Inventory Visibility
- Logistics Management
- Reverse Logistics

1.4.4 Supply Chain Event Management

The Event Management product suite is a focused solution for Supply Chain Event Management, that leverages the capabilities of the Application Platform (process configuration, event engine, status monitoring) and Analytics and the exception console for comprehensive supply chain visibility (orders, inventory, shipment, returns, etc.) and exception handling.

The Event Management product suite provides fully integrated event configuration, status and event monitoring and alert handling capabilities. It provides the underlying mechanism for setting and monitoring conditions, or "events", that drive transactional activity within the critical supply chain processes, such as fulfillment, inventory management and purchasing. When deployed in conjunction with Selling and Fulfillment Foundation, the event engine enables processes to be modeled and managed based upon events occurring (such as a "pull signal" from field sales to begin manufacture of a customer's product, a reminder about a scheduled release on a contract, etc.) rather than within a pre-defined, hard-coded application procedure. Exceptions can be handled automatically as well as through configurable exception consoles with full tracking, automatic escalation and resolution with complete audit history.

The Event Management product suite includes the Application Platform and Supply Chain Event Management licensed products together with the applications necessary for the targeted supply chain process, such as Distributed Order Management for order visibility, Global Inventory Visibility for inventory visibility, and so forth. It is, in fact, the link with our applications that allows us to differentiate our Supply Chain Event Management, by enabling the "control", not just monitoring and notification, of critical processes.

1.5 Selling and Fulfillment Foundation Documentation

When installing or upgrading Selling and Fulfillment Foundation, you can select whether you want to enable an Online Documentation Library or a Local Documentation Library of the product documentation. You can also change this selection after installation through the ACTIVE_HELP_URL property in sandbox.cfg, as described in the *Selling and Fulfillment Foundation: Properties Guide*.

The Online and Local Documentation Libraries offer the following features.

- Online Documentation Library - This library, hosted by Sterling Commerce, provides:
  - Online access to the Selling and Fulfillment Foundation documentation set in HTML and PDF formats
  - Google mini-search capability across the documentation set
  - Dynamic updates to the documentation for changes and hotfixes
- Local Documentation Library - This library, hosted locally by your enterprise, provides:
  - Local access to the Selling and Fulfillment Foundation documentation set in HTML format
  - Index and search capability on a book-by-book basis
  - No updates for changes and hotfixes except through product upgrades

Note: The Online and Local Documentation Libraries are different from the Context-Sensitive Help that you can access from the Help button in the user interface. The Context-Sensitive Help provides a single page of help documentation, while the Online and Local Documentation Libraries provide the entire documentation set.

For more information about enabling Context-Sensitive Help and the Documentation Libraries, see the *Selling and Fulfillment Foundation: Installation Guide* and the *Selling and Fulfillment Foundation: Properties Guide*.

2

Application Platform Technology Concepts

The Application Platform is the technical foundation and framework that supports and enables the smooth flow of business transactions.

It uses the latest technologies and standards that enable interoperability, including Java, Enterprise Java Beans (EJB), Java Messaging Services (JMS), Java Management Extensions (JMX) and Extensible Markup Language (XML). Selling and Fulfillment Foundation is designed to provide the best in flexibility for deployment in complex environments and adaptability to changing business processes. It provides the framework that enable companies to do business in an extended enterprise environment.

2.1 The Application Platform Framework

The Application Platform framework enables you to deploy an enterprise wide application that conforms to scalability and high-availability requirements. It provides tools for effective monitoring and management of application components. It provides the capabilities to manage the extended enterprise network with critical security, internationalization and localization, and system management capabilities.

The application framework also allows the extension of Selling and Fulfillment Foundation data model to support unique data needs of your implementation.

You can extend Selling and Fulfillment Foundation by changing the on-screen elements of the user interfaces, modifying printed documents, or adding columns to the database tables.

2.2 User Interface Extensibility

User Interface extensibility allows you to change the way information is rendered, or displayed, without changing the way it functions. You can customize the look and feel of the Application Console and the Applications Manager which make up the Selling and Fulfillment Foundation user interface:

- Application Console - the standard GUI for creating, tracking, and viewing orders, item inventory, and returns
- Applications Manager - the graphical user interface for configuring Selling and Fulfillment Foundation

This is accomplished through a combination of configuration changes through the Applications Manager and HTML code changes to the JSP files.

Additionally, you can extend the look and feel of Printed documents.

2.2.1 Application Consoles User Interface Extensibility

The Selling and Fulfillment Foundation Presentation Framework allows you to extend the user interface of the standard Application Consoles. The standard Application Consoles user interface uses HTML embedded within Java Server Pages. The UI layer goes through the Service Definition Framework to access the APIs, which ensures that only exposed APIs are used.

The user interface layer of the Service Definition Framework uses very minimal XML manipulation. Wherever significant manipulation of XML output becomes necessary, changes to the APIs provide a more user interface friendly output.

The Selling and Fulfillment Foundation Presentation Framework provides you with the capabilities to extend the standard Application Consoles user interface in the following ways:

- Customize the Selling and Fulfillment Foundation Sign In screen - the first page a user sees when they start Selling and Fulfillment Foundation - in the following ways:
  - Set the locale setting to determine the language in which on-screen literals are displayed

  - Configure locale-specific logins - for multilingual user communities, it makes sense to display the login page in every possible language
  - Establish a corporate look and feel - a secondary purpose of the Sign In screen is to introduce the corporate look and feel that follows throughout the application
  - Sign in from an external application - when integrating Selling and Fulfillment Foundation with external applications, you can automatically log a user into Selling and Fulfillment Foundation from the external application
  - Support single sign-on - users can log into Selling and Fulfillment Foundation transparently using a domain password. Selling and Fulfillment Foundation supports third-party single sign-on applications
- Add corporate branding - you can change the branding logos displayed on screen in the following locations:
  - Login Screen (Sign In Window)
  - Menu Bar
  - About Box
- Define a theme - a theme defines the look and feel, for example fonts and colors, of the application
- Customize views - screen layout and organization for the following types of views:
  - Search Views
  - List Views
  - Detail Views
- Customize business entities
- Add lookups - you can add lookups that allow users to choose from an assortment of options rather than typing in data
- Add graphs and pie charts - graphs and pie charts allow users to view a graphical representation of data

- Customize menu structure - When creating customized screens, you need make sure that users can access them, either through a menu structure or through navigation
- Customize the Selling and Fulfillment Foundation images - you can customize the Images used throughout Selling and Fulfillment Foundation which appear in the following places:
  - Menu bar
  - Menu
  - Application Console
- Customize screen navigation - you can customize how users navigate from entity to entity by configuring link or action resources
- Create custom event handlers - you can create and plug in custom client-side validations to user interface controls at control- or screen-level
- Create screens for custom document types
- Create custom transactions - you can create custom order or delivery transactions and configure the pipelines to include these transactions to do status changes
- XML binding - developers can easily form the input necessary to pass to an API and populate a screen with the output of the API

2.2.2 Applications Manager Extensibility

The Applications Manager user interface was developed in Swing. The main purpose of user interface extensibility is to allow any database extensions to be integrated into the graphical user interface.

Extensibility includes the following modifications:

- Adding any buttons and labels
- Adding any text fields and check boxes -
- Hiding any non-mandatory components
- Reorganizing the components that are displayed on-screen
- Creating or modifying user themes

You may modify the following types of screens:

- Search screens
- Detail screens
- List screens - you can add, remove, and re-arrange columns in the list screen within the Applications Manager.

Screens that contain only a Tree structure are not extensible.

2.3 Database Extensibility

Database extensibility enables you to add columns to the Selling and Fulfillment Foundation tables to capture and maintain additional data. You can extend certain Selling and Fulfillment Foundation tables by adding one or more columns to the table.

Additionally, you may encounter a situation where you have dynamically changing fields for special orders. Selling and Fulfillment Foundation enables you to extend order- and line-level information that is captured on the system by adding order or order line-related attributes. This allows you to capture company-specific flat and hierarchical data. Selling and Fulfillment Foundation APIs transparently provide access to such data to ensure that all changes to the data model are automatically integrated into Selling and Fulfillment Foundation. These attributes do not appear in the default user interface and are not searchable. However, you may customize the user interface to include such fields.

2.4 Security Management

Security Management enables you to ensure that each user accesses only the information that is appropriate for carrying out their tasks. the resources provided by the organization to which they belong. A user is limited to access only those resources to which they have permission.

For more information about installation level security, see the *Selling and Fulfillment Foundation: Installation Guide*.

Users

A *user* is a single person assigned with a certain task, such as, Hub Administrator or Customer Service Representative (CSR), depending on

what role they play in the organization. Each organization has its own users.

User Groups

A *user group* is a collection of users who perform a similar task. For example, a group of Customer Service Representatives might be collectively placed in a CSR user group. Users can belong to multiple user groups. Permissions are assigned to a user group. A user retains all permissions for each user group to which he or she belongs.

Each organization has its own user groups. User groups can only contain users for the same organization of which the user was created, except in the case of a user group created by the Hub organization, which can contain users of any organization.

Teams

A team is a collection of users who have common data access requirements. Teams can have access to specific document types, Enterprises, ship nodes, and customers. If a user is not associated with a team, that user is considered to have the least restrictive access, or default access. By defining a team, you can further restrict the access to any Enterprises, document types, or ship nodes that are a sub-set of the default access list. For more information about teams and how you can use them in Selling and Fulfillment Foundation, see Section 3.2.2.5, "Customer Management".

Resources and Resource Permissions

Selling and Fulfillment Foundation consists of many ***resources***, including screens, functions, URLs, and so on, for which permissions can be granted or revoked.

Through Security Management user groups are granted access rights to predetermined Resources. A user is limited to access only those resources that have been permitted to at least one of the user groups of which the user is a member.

2.5 Security of Sensitive Information in Log Files

Selling and Fulfillment Foundation provides a default log filter that enables you to mask sensitive information, such as CVV2 codes, in log files.

For information about how to mask sensitive information in log files, refer to the *Selling and Fulfillment Foundation: Properties Guide*.

2.6 Data Access Policies

Selling and Fulfillment Foundation provides the ability to control user-specific access to data, such as orders and shipments, through global rules configuration.

> **Note:** If you want to use your own access rules, Selling and Fulfillment Foundation provides the YSCGetExternalDataAccessPolicyOnReadUE and YSCShouldAccessPolicybeAppliedOnUpdateUE user exits. For more information about these user exits, refer to the *Selling and Fulfillment Foundation: Javadocs*.

Selling and Fulfillment Foundation access rules are applied via the following access policy modes:

- Enterprise User -this user belongs to the Enterprise that brokers business. Each Enterprise can contain multiple organizations that are assigned various roles.
- Buyer User - this user belongs to the organization that purchases products from an Enterprise or other seller organizations.
- Seller User - this user belongs to the organization that supplies products to the Enterprise or other buyer organizations.
- Node User - this user belongs to an organization that represents a physical location, whether it is a manufacturing plant, small stock room, or warehouse.

In addition to user access policy modes, Selling and Fulfillment Foundation controls data access by applying catalog access policies and pricing access policies.

- Catalog Access Policies

  Catalog access policies are applied when the user *manages* catalog data, such as items, categories, and attributes. If the user is only *viewing* catalog data, catalog access policies are not applied.

  Catalog access policies will only be applied if the user is an enterprise user or a catalog organization user.

  A catalog organization user can manage data for the user's own organization or for the organizations administered by the user's organization. However, a catalog organization user cannot manage data for subcatalog organizations. Similarly, a subcatalog organization user can manage data for the user's own organization or for the organizations administered by the user's organization.

  For example, if an item is owned by a catalog organization (that is, there is no subcatalog organization on the item), the item cannot be managed by a subcatalog organization user. However, if an item is owned by a subcatalog organization (that is, there is a subcatalog organization on the item), it can only be managed by a subcatalog organization user.

  Item entitlements can also be applied to access policies based on the purpose. For item entitlement, access policies still get applied, even if the user is only *viewing* item entitlements. For example, buying entitlements (purpose is buying) can only be viewed and managed by an enterprise user. Selling entitlements (purpose is selling) can only be viewed and managed by a catalog organization user.

- Pricing Access Policies

  Pricing access policies will only be applied if the user is an enterprise user, a seller, or a pricing organization user. Any user playing these roles can manage pricing data for the user's own organization or for the organizations administered by the user's organization.

  Users can see all price lists defined by their pricing organization, but they can modify only the price lists of their own pricing organization. For example, if XYZ-MA and XYZ-CA use XYZ-MA as a pricing

organization, an XYZ-CA user can see all price lists defined by XYZ-MA, but only users of XYZ-MA can modify those price lists.

Note: Access policies are applied to transactional and master data only. Access policies are not applied to configuration data, such as organization and users.

A user may log into an application in any of the above user roles. For example, a user could log in as a buyer to look at sales orders, or as a seller to look at purchase orders. You could configure Enterprise User access for an internal storefront administrator, or Node User access for a warehouse administrator who wants to see all the nodes in the organization.

Each data access mode offers flexible levels of user access, such as access to all customers assigned to a team or only to customers directly assigned to that user. Access can also be extended to descendant teams and descendant customers, depending on the access mode.

Figure 2–1 is an example of a business that does not want to assign CSRs to specific accounts, but rather assigns entire teams to accounts. For more information about teams, see "Teams" on page 2-16.

Figure 2–1 Data Access Team Account Scenario

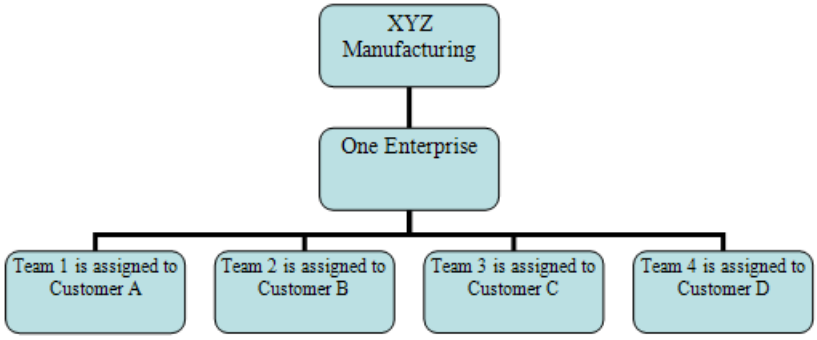


Figure 2–2 shows an example of a business with two enterprises: one for retail and one for small-to-medium businesses. These two enterprises

have separate CSR teams. They mix their use of the data access features by letting the retail enterprise, E1, see any enterprise, while the other enterprise, E2, must assign its CSRs to particular accounts.

Figure 2–2 Data Access Mixed-Mode Scenario

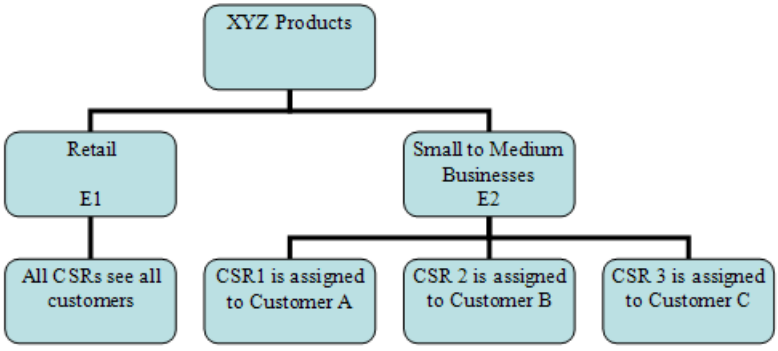
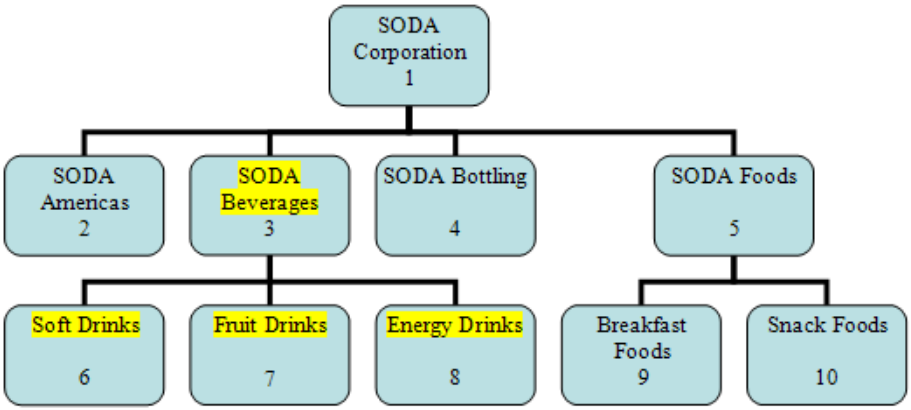


Figure 2–3 shows the organizational structure of a soft drink business that is buying telephony equipment from HIJ Telephone Corporation. In this Buyer scenario, The SODA Beverages Division (3) can buy telephone equipment for itself and its descendent organizations, Soft Drinks (6), Fruit Drinks (7), and Energy Drinks (8). Alternatively, the Data Access features could be used to exclude one of the descendants, Energy Drinks (8) from the organizations that the SODA Beverages Division (3) can buy telephones for.

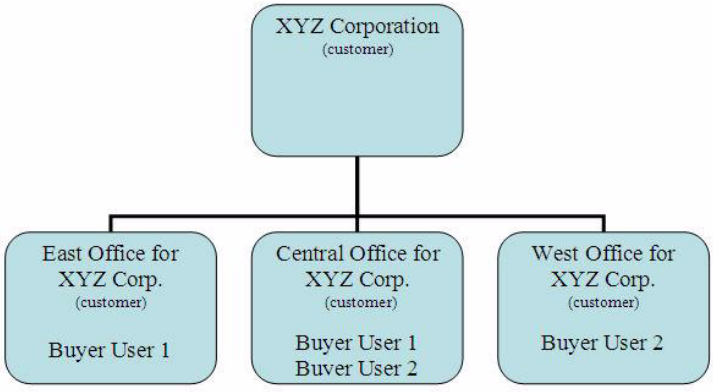
Figure 2–3 Data Access Buyer Scenario



You can configure all data access policies in the Application Platform of the Applications Manager.

Figure 2–4 shows the organizational structure of XYZ Corporation, which is a customer of an office supply company. The East, Central, and West offices are child organizations of XYZ Corporation and are also customers of the office supply company. In this buyer scenario, Buyer User 1 is assigned to the East and West offices, and Buyer User 2 is assigned to the Central and West offices. Buyer user access policies are configured to provide buyer users with access to assigned customers only. Thus, Buyer 1 can purchase supplies for the East and Central offices, and Buyer User 2 can purchase supplies for the Central and West offices.

Figure 2–4 Data Access Buyer Assignment Scenario



You can configure all data access policies in the Application Platform of the Applications Manager.

2.7 Password Policies

Selling and Fulfillment Foundation provides a built-in, flexible password management policy for controlling password use and behavior. A password policy is a set of rules to define, control, and manage user passwords. Although a set of default rules is provided, you can configure your own password policy rules.

The password policy broadly governs the following:

- Password strength - This includes the length (minimum and maximum) of the password, special characters in the password, and password reuse.
- Password generation - This includes generating a password during user creation, frequency of password expiration, failed login attempts, and user roles that may affect the password policy.
- Password reset - This includes resetting of passwords through different protocols, such as e-mail or SMS.

In addition, password policy configuration denies access to users (locks them out) in case of repeated invalid login attempts.

For additional information about password policies, see the *Password Policy Management* guide.

2.8 Password Encryption

Passwords in the Selling and Fulfillment Foundation database are automatically protected from potential attackers by a secure hash algorithm.

The system protects users' passwords by first appending a string of random characters (salt) to each password. The salted password is then hashed and stored in the database. The system stores SHA-256 hash with 16 bytes of salt (random data) for each password.

The YCPValidateChangedPasswordUE and YCPCheckPasswordsMatchUE user exits can be implemented to provide customized hashing logic.

2.9 JMS Security for JBoss

Securing the JBoss JMS Queue provides security to the data stored in queues on application servers. For JBoss, Selling and Fulfillment Foundation provides only JMS Queue-based security. While connecting to a secured JMS Queue, you will need to pass the username and password for authentication.

You can configure JMS security for JBoss by setting up the JBoss Default Messaging JMS Queue transport node available in the Applications Manager. For more information about JBoss Default Messaging JMS Queue node, see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide*.

You should also secure the JMS queue on the JBoss application server.

2.10 API Security

Selling and Fulfillment Foundation provides authentication for API calls for a given user ID, a certificate, or both. It allows you to control what API resources can be accessed when an API is called.

When calling an API, you can provide following two levels of security:

- Authentication with a user ID, a certificate or both. The login API is called before any other API is called. The user authentication is done

prior to API call. The access can be restricted to a subset of APIs or Services for a given user.

- Authorization, which verifies which resources you can access. The authorization restricts the user to a subset of APIs and Services. This authorization is done at the API level for all APIs, regardless of where that API is invoked from. This authorization is completely independent of UI resource security. For example, say a user has access through the UI to a screen that lists users. The screen invokes the `getUserList` API. In order for the screen to work for that user, they must have access to the `getUserList` API.

For more information on how to configure API security, refer to *Selling and Fulfillment Foundation: Application Platform Configuration Guide*.

2.11 System Management

In Selling and Fulfillment Foundation you can manage integration and agent servers, view the properties of your application servers, enable database caching, and increase trace log output for APIs, user exits, services, and agents.

The System Management module provides features to administer and monitor various components that make up Selling and Fulfillment Foundation. The System Management Console provides a complete picture of Selling and Fulfillment Foundation while it is running. Additionally, the health monitor agent can alert system administrators when a problem happens such as an application server going down or an agent server not processing tasks.

Other features include the following:

- View real-time statistical information to understand what is happening in Selling and Fulfillment Foundation at any moment
- Monitor application servers for status and response time.
- Monitor APIs for response time
- Monitor agents for status and number of pending tasks to ensure a bottleneck is not created
- Monitor JMS queues to ensure messages are being processed properly
- Shutdown, suspend, or resume agent and integration servers

- Clear database cache
- Trace APIs, agents, user exits, services, and the Application Consoles
- Collect and persist statistical data

2.12 Queue Management

Queue Management enables the configuration of queues to notify users when alerts are raised by the system. These alerts can come in different formats including e-mail, faxes, and so on. Alerts are sent to different queues depending on the type of alert.

An ***alert*** is a message directed to a user or queue about a transaction that needs manual intervention.

Queues are configured to distribute alerts to users. You determine which users receive different alert types by assigning them to queues. You can also set up alert priority and actions raised when certain conditions are met for the alert.

In Selling and Fulfillment Foundation you can manage alerts, set the priority for the alerts, enable the system to automatically assign an alert to a user, classify the alerts as one or more types, route the alert to a specific queue by alert type, add notes to an alert, and specify a context sensitive resolution screen for the alert.

You can enable the system to automatically assign an alert to a user instead of someone having to go and manually assign an alert. You can classify alerts as one or more types and use these types to:

- Default attributes (like default follow up time and priority) onto an alert.
- Specify a context sensitive resolution screen for the alert.

You can specify the priority, queue, expiration days, follow-up hours, resolution form, and user for the various alerts types in the configuring alerts component of the Applications Manager. For more information about configuring alerts service, see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide*.

2.12.1 Alert Consolidation

Alerts can be raised under different situations and in a number of circumstances the content of the new raised alert is the same as an existing alert. For example, a monitor triggers an alert based on an order condition. The order does not change for a certain duration and another alert is raised. Instead of having numerous identical alerts you can consolidate the alerts using the alert consolidation method.

You can enable alert consolidation in the Service Definition Framework Alert component. This consolidation is available everywhere regardless of where the alert is raised, including APIs, service builders and error handling. For more information about enabling alert consolidation, see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide*.

When an alert is raised and consolidation is enabled, the database is queried to find any other alerts with identical information. If such an alert exists a consolidation count is incremented on the existing alert and new record is not inserted into the database.

To identify the attributes used for consolidation, a consolidation template can be passed as part of the input to the API being called. This template contains all attributes that must be the same in the input XML and the database before consolidation takes place. If a template is not passed, then a default template is used, consisting of many of the Inbox attributes and all of the extended attributes. This default template can be found in `<INSTALL_DIR>/repository/xapi/template/merged/resource/exceptionConsolidation.xml`, and can be extended in the same way as API templates. For more information about extending the API templates, see the *Selling and Fulfillment Foundation: Extending the Database Guide*.

A consolidation window is set to consolidate alerts by the `Day` or the `Hour`. When the date attributes in the consolidation template are compared with the database values this consolidation window is used to determine whether they are equal. The consolidation window matches a record within a discrete time interval and are outlined below with an example.

Consolidation by Hour

For example, consider that the consolidation window is set to `HOUR` and the timestamp of the last occurred alert is 2005-02-01, 06:24 AM. A

similar alert is raised on 2005-02-01, 7:10 AM. This new alert raised at 7:10 AM is not consolidated with the alert raised at 6:24 AM even though the alerts are less than an hour apart. Hence for consolidating these alerts, they must occur within the same hour.

Consolidation by Day

For example, consider the consolidation window is being set to DAY and the last occurred alert was on 2005-02-01, 07:40 PM. Even though a new alert is raised at 2005-02-02 1:30 AM, this is not consolidated with the previous alert since they were raised on different days.

> **Note:** If the consolidation template has only non-time attributes then the consolidation window (hourly or daily) would have no effect on the consolidation process.

The consolidation window is applied to all timestamp fields that are included in the consolidation template including any extended fields.

The following are the assumptions and limitations of consolidating the alerts:

- If the alerts are consolidated, no alert attribute fields are updated other than the consolidation count and the last occurred time of the alert.
- If there are multiple alerts that match the same consolidation template the exact alert that get chosen for matching is undefined.
- Alerts cannot be consolidated on any child tables or extended hang-off tables.
- You cannot enable conditional consolidation based on the data. For example, you cannot choose to consolidate only when `EnterpriseKey` is DEFAULT.
- Closed alerts are not used for consolidation.

You can specify the expiration days for the alerts in the alert service of the Applications Manager. The expiration days represent the number of days of no activity after which this exception may be automatically closed. A value of zero means the exception does not expire.

2.13 Print Extensibility

The Sterling Warehouse Management System allows you to create custom labels and prints by configuring with Loftware Print Server and Loftware Label Manager systems. The print services are customized by creating service definition to be invoked from an event or the console user interface. The standard labels provided with the Sterling WMS can be customized by integrating with Loftware Label Manager.

For performing such customizations, refer to the *Selling and Fulfillment Foundation: Customizing APIs Guide*.

2.14 Mobile Application Extensibility

The Mobile Application Extensibility enables you to develop and display custom user interface for the mobile devices used in warehouse operations.

You must prepare the development environment for deploying and testing your customizations of the Selling and Fulfillment Foundation mobile user interface. For procedures on customizing the mobile application interfaces, refer to the *Selling and Fulfillment Foundation: Customizing User Interfaces for Mobile Devices Guide*.

2.15 Alert Management

In Selling and Fulfillment Foundation you can:

- Configure alerts
- Create new exception types
- Set the priority for the exception types
- Classify each exception as one or more types
- Route an exception type to a specific alert queue
- Specify the threshold for high priority alerts
- Specify the expiration days and follow-up hours for an exception type
- Specify whether or not to consolidate exceptions of the type

You can classify exceptions as one or more types and use these types to:

- Default attributes (such as default follow-up time and priority) for an exception type
- Specify a context-sensitive resolution screen for an exception type
- Specify a configuration screen for an exception type
- Specify a list screen for an exception type

You can define additional configurations for an exception type. For example, PCAs may want to capture additional information such as Credit Limit for which they can specify the configuration form used to capture the required information. PCAs can also specify a list form, which provides a list of alerts defined for a specific exception type. You can also associate exception types to an application, such as a PCA, and a role, such as BUYER or SELLER, that an application supports at the HUB level.

You can configure one or more exception types and its priority for an organization. You can route the exception to the appropriate alert queue by configuring a routing rule for the exception type. You can also create one or more routing rules for each exception type configured for an organization. When creating routing rules for an organization level exception type, you can specify the routing logic that you want to use to raise an alert. More than one alert may be raised based on the routing logic used. The routing logic determines the queue to which the exception needs to be assigned. The routing rules indicate the type of routing. For a given exception type, you can use a different type of routing rule or logic.

Table 2–1 The following routing logic is provided out of the box

Routing Type	Description
Use Department's Queue	Use the default queue of the department.
Use Specified Queue	Use the queue specified in the routing rule.
Use Interface to Obtain Queue	Use a Java class to obtain the queue.
Route to Role	Use the routing rule of the organization whose role you specified. (Buyer, Carrier, Enterprise, Hub, Node, Seller)

Table 2–1 The following routing logic is provided out of the box

Routing Type	Description
Route to Custom Role	Use a custom organization attribute to determine the queue.
Route to Exception Type's Queue	Use the queue specified on the exception type.

When creating routing rules for an organization-level exception type, you can also specify whether or not to send an e-mail to the specified e-mail address. If the routing type "Use Department's Queue" is used, for which e-mail needs to be sent, and if an e-mail ID is not specified in the exception routing rule, the system checks the configuration of the department for an e-mail ID. When an exception is created on a default queue of the department, an e-mail is sent to the e-mail configured for the department. If the template is not configured, a default template is used.

For more information about configuring alerts, see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide*.

3

Participant Management

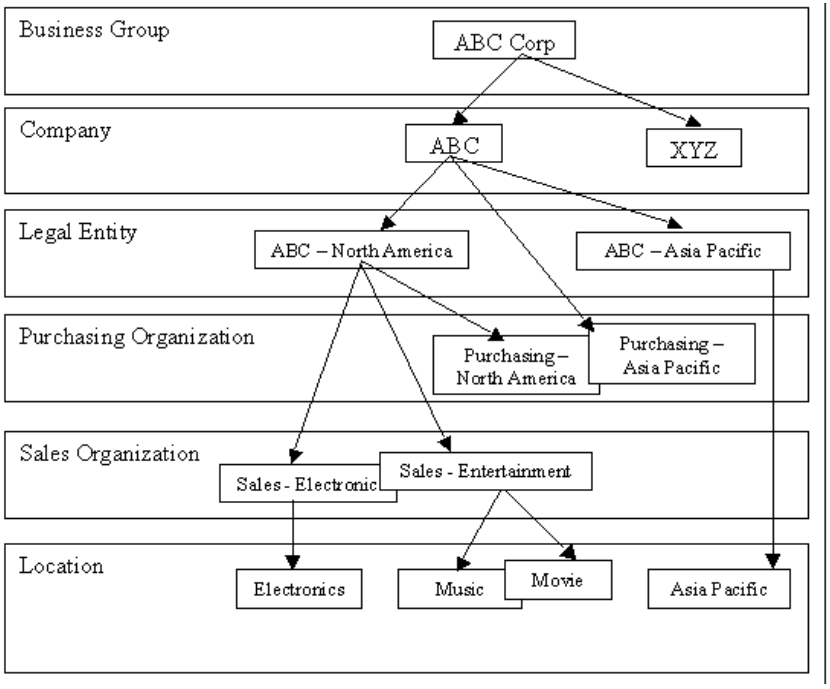
3.1 Organization Modeling

An *organization* represents a company, department, cost center, division, sales unit or any other organizational unit within a business. Typical organizations are as follows:

- Business Group
- Company
- Legal Entity
- Sales Organization
- Purchasing Organization
- Plant and Warehouse

Figure 3–1, "Typical Organization Model" depicts a typical organization model.

Figure 3–1 Typical Organization Model



These organizations are typically described as follows:

Business Group

A *business group* is typically the highest level in the organization hierarchy. It has no accounting impact. It consists of one or more companies.

Company

A *company* typically represents a global brand name and is the organizational unit for which individual financial statements are created according to the relevant legal requirements. A company may have one or more legal entities.

Legal Entity

A *legal entity* represents an organization unit identified by local governments as operating units and are typically instituted for every country a business operates in. This organizational unit is typically self-contained and set of accounts can be drawn up for external reporting. This involves recording all relevant transactions and generating all supporting documents for financial statements, such as balance sheets and profit and loss statements.

Sales Organization

A *sales organization* is responsible for sales and distribution of products and services. Sales organizations can be defined based on the following characteristics:

- Sales channel **-** for example, wholesale, retail or direct sale.
- Product Line **-** for example, electronics, entertainment, and service.
- Geography **-** the geography of the ship-to location of orders. For example, east-coast or west-coast.
- Customer **-** (major customers).
- A combination of one or more of the above.

Purchasing Organization

A *purchasing organization* (also known as a Buyer organization) is responsible for placing purchase orders to *vendors* to replenish raw materials and products in a company's locations. Purchasing organizations could be created centrally or can be associated with each legal entity or sales organization. Purchasing organizations can also be modeled based on product lines, geography, or vendors.

Plant and Warehouse

Plants and warehouses are physical locations where goods are manufactured or stored for distribution. Typically, a plant or a warehouse is owned by a legal entity. A business can have many plants or warehouses associated in almost all combinations to its sales and purchasing organizations.

A warehouse could be designated to store products for a single sales organization or could service multiple sales organizations based on the sales organization structures (product-based, geography-based,

customer-based). A single purchasing organization may be responsible for buying raw materials for a plant or stocking product into a warehouse or multiple purchasing organizations could be involved with a single plant or warehouse (purchasing based on geography or product).

In a third-partly logistics (3PL) company or in some rare cases with a business, a specific location can store inventory for multiple legal entities or companies. In this case, typically, the 3PL company is the owner of the inventory stocking location.

3.2 Organization Modeling in Selling and Fulfillment Foundation

In Selling and Fulfillment Foundation, an "organization" represents any unit of a business whether it is a company, legal entity, a business group, sales organization, purchasing organization or warehouse. Departments and divisions would also be represented as an organization in Selling and Fulfillment Foundation. Organizations can be defined in Selling and Fulfillment Foundation in a hierarchical structure. Organization hierarchies are used in multiple places in Selling and Fulfillment Foundation to inherit configurations and other business rules. Even customers and vendors are defined as multi-level organizations. Section 3.2.2, "Customers and Vendors" discusses customer and vendor modeling in Selling and Fulfillment Foundation.

When defining Participants of your organizational model in Selling and Fulfillment Foundation, in addition to assigning organization roles, any organization in can also be designated as having the following responsibilities:

- Enterprise
- Catalog Organization
- Inventory Organization
- Capacity Organization

An organization can be defined with any or all of the above responsibilities at the same time.

Enterprise

An Enterprise represents the organization that owns and controls all transactions in Selling and Fulfillment Foundation. An Enterprise in Selling and Fulfillment Foundation controls the flow of documents (such as a sales order) and is considered the owner of the document. Most business rules and fulfillment processes for an order is defined by the enterprise. On a sales order, the Enterprise is also assigned the role of the Seller organization in most cases. Similarly, for purchase orders, an Enterprise is also assigned the role of the Buyer organization in most cases. In some cases, if a higher level organizational unit wants to control and enforce business rules or document flow of all its subsidiaries, that organizational unit is assigned an Enterprise role and its subsidiary organizations are assigned Seller and Buyer roles.

Even though most business rules are controlled by the Enterprise, pricing rules are always controlled by the seller organization in both sales and purchase situations.

In an organization hierarchy, multiple organizations can be designated as an Enterprise. However, every organization in the organizational structure needs to either be defined as an Enterprise or have an organization designated as its primary Enterprise. This is necessary so that at all times Selling and Fulfillment Foundation can clearly identify the set of rules to be applied to this organization. (Even though this primary Enterprise relationship is established, an organization can participate with another Enterprises on an individual transaction basis).

Every transactional document (order, shipment, load) in Selling and Fulfillment Foundation requires a designated Enterprise.

Catalog Organization

A Catalog Organization represents an organization level in the organization hierarchy at which an item master is defined. A catalog organization provides any of the following functions:

- Item master definition. Even in a multi-level organizational structure, an item master is typically defined at a very high level in the hierarchy. This allows all other organizations to share one common item master definition; eliminating the need to create a separate definition of their own.
- A mechanism to separate item master definitions of two distinct organizations. For example a 3PL scenario and companies under the

business group acquired through a merger and acquisition (M&A). Same product identifiers could represent completely separate physical products across two catalog organizations.

- A mechanism to cross-reference product identifiers between two catalog organizations through usage of Global Trade Identification Number (GTIN).

Catalog Organizations can have one or more catalogs (such as master catalogs). However, an item can be defined in only one master catalog. A master catalog represents a product line in most scenarios. In organizations where divisions are created based solely on product, a master catalog could represent a division.

Typically the Catalog Organization is designated at the company level.

Inventory Organization

An Inventory Organization represents an organization level in the organization hierarchy at which all inventory information is consolidated. An Inventory Organization provides any of the following functions:

- Inventory identification for a product. Different organizations could have different product identifiers for the same inventory item. Inventory organization provides a mechanism to rationalize these product identifications into a single nomenclature across multiple organizations. This allows a consistent global view of a product across organization hierarchies and enables better decision making for the business.
- Establishes ownership of inventory when a single physical location is shared across multiple organizations without having the need to create multiple logical locations to establish the inventory ownership (for example, 3PL scenarios)
- Inventory separation. All organizations that are defined as part of the same Inventory Organization have visibility to inventory of all other organizations that are part of this inventory organization. This allows better visibility across organizations but can also create data security and other issues. An Inventory Organization creates a separate silo of inventory definitions in the system. This silo cannot be accessed by organizations belonging to a different silo and thus provide a complete segregation of inventory.

All organizations must either be defined as an Inventory Organization or must designate another organization as their Inventory Organization. An Inventory Organization designated for an organization should share the same Catalog Organization. This ensures that appropriate cross-referencing can be made to arrive at the inventory identification of an item. This also ensures that two separate physical items do not create a situation where they have the same inventory identifier within the inventory module and thus can't be differentiated.

Also note that, if the sales organization legal entity is not the same legal entity associated with the physical location from where the product is being sourced, Selling and Fulfillment Foundation has the capability to automatically generate a purchase order (created as chained order) to ensure proper inventory ownership transfers. This is done even though the Inventory Organization is common. In some cases, this is not desirable - as in the case of a 3PL where the physical location is owned by the 3PL organization and does not indicate real inventory ownership. This can be prevented in one of the following two ways:

- Designate a node as a "3PL" node to suppress such chained order creations.
- Flag the owner organization of the node as "chained order not required".

Typically, the legal entity in an organization hierarchy is designated as the Inventory Organization for all subsidiary organizations.

Capacity Organization

A Capacity Organization represents an organization level in the organization hierarchy at which all capacity information is consolidated. A capacity organization provides any of the following functions:

- Defines service slots for order promising functions.
- Capacity separation. All organizations that are part of the same Capacity Organization have visibility to the capacity of all resources that are part of this Capacity Organization. This allows a better visibility across resources but can also create data security and other issues. A Capacity Organization creates a separate silo of capacity definition in the system. This silo cannot be accessed by organizations belonging to a different silo and thus provide a complete segregation of capacity.

3.2.1 Enterprise Onboarding

The Application Platform framework provides the ability to bring new enterprises on board the existing configuration. It also helps you separate the existing organizations from newer organizations, and reduces the incidence of various organizations interfering with each other.

As part of enterprise onboarding, you can separate configurable resources such as templates, user exits, and so on, based on an enterprise. You can also add new enterprises without impacting the existing enterprise resources. The resources configured at the enterprise level are identified using a unique Resource Identifier, which can be used to locate the resources belonging to the corresponding enterprise.

Enterprise Inheritance

As part of the enterprise onboarding feature, the Application Platform provides an enterprise with the capability to inherit the configuration from some other enterprise, for example, enterprise B can choose to inherit the configuration from an existing enterprise A, instead of defining an entirely new set of configurations.

For more information about enterprise inheritance, see the section Determining Inheritance in the *Selling and Fulfillment Foundation: Application Platform Configuration Guide*.

Packaging Extensions

In a multi-enterprise setup, managing various levels of extensions and configurations can be cumbersome. For example, as part of enterprise onboarding, if you want to expose an additional attribute, you must modify the default template to include the additional attribute. This could have an impact on the other on-boarded enterprises. To resolve this complexity, the Application Platform provides the ability to develop and package enterprise-specific resources such as database extensions, templates, UI resources, and so on as an extensions service package. This package contains all the components to on board an enterprise.

For more information about developing and packaging enterprise-level resources, see the section Building and Deploying Enterprise-Level Extensions in the *Selling and Fulfillment Foundation: Customization Basics Guide*.

Multitenant and Multischema Architecture

The multitenant architecture of Selling and Fulfillment Foundation provides the ability to support multiple enterprises (tenants) on a single deployment of the application and to segregate data and configurations accordingly. This allows you to leverage your investments in application servers and hardware.

A multitenant deployment consists of multiple enterprises that have unique business needs, such as different process flows, enterprise specific extensions, and rules. In multitenancy, data is partitioned into Transaction and Configuration schemas so that each customer works with a customized virtual application instance.

A multischema deployments is a type of multitenant deployment that has multiple transactional database schemas.

Multischema deployments comprise the following schemas:

- Configuration - Stores system and business rules, as well as organizations.
- Transaction - Stores data transactions related to orders, shipment tasks, inventory.
- Master Data - Residing along with the Transaction schema, contains data created through batch feeds, such as items and customers.
- Metadata - Contains data required for database lookup of configuration information.
- Statistics - Maintains statistics for the application, such as agent and API statistics.

In multischema deployments, one enterprise can be upgraded at a time, while others remain on the same version until ready to upgrade.

You can enable multischema mode when installing Selling and Fulfillment Foundation, or install in single-schema mode and enable multischema mode later. If you are upgrading to Release 9.0, you can enable multischema mode after upgrade, since upgrades can be done only in single-schema mode.

For more information about multitenant and multischema deployment options, benefits, and limitations, see the *Selling and Fulfillment Foundation: Multitenant Enterprise Guide*. This guide also explains multischema concepts in depth and how to enable multischema mode.

3.2.2 Customers and Vendors

Any business group can be represented in Selling and Fulfillment Foundation as an organization with all its subsidiaries defined in a hierarchical fashion. Any participant viewing this structure would see it in the same consistent way.

Selling and Fulfillment Foundation models customers and vendors as a relationship between two participants. In this relationship, organization X can have its own distinct identifier (Organization ID) for organization Y when it models organization Y as its customer and another identifier when it models organization Y as its vendor. This becomes especially true when multiple Enterprise Resource Planning (ERP) systems are involved with each one creating it's own identifier. Even though Selling and Fulfillment Foundation provides strong rationalization capabilities to model a participant as a single organization playing multiple roles, they recognize that most other supply chain systems do not yet have such strong capabilities.

3.2.2.1 Customer and Customer Contact Definition

When defining customers, Selling and Fulfillment Foundation ensures that the customers be modeled as organizations. Selling and Fulfillment Foundation provides the flexibility to use the same organization identifier as the customer identifier *OR* to generate or input a different identification code. A customer creation API is enabled to automatically create the organization when a customer is created.

The default billing address, shipping address, sold to address, and payment method are automatically picked up by the Selling and Fulfillment Foundation order entry function or order creation API. This provides you with the capability of modeling the ERP in Selling and Fulfillment Foundation and the continued ability to rationalize organizations across ERP systems into one common organization to allow better customer service and decision support.

A rule can be set to determine whether the default addresses and payment method in an order are pulled from the customer or the customer’s parent in the customer hierarchy. Also, the addresses and payment method on the order can be defaulted from a specific customer contact if valid customer contact information is passed as part of the order entry. A customer contact is an individual consumer in a B2C

relationship or an individual contact within a customer enterprise in a B2B relationship.

Any Enterprise can create Customer entities as needed. Customer roles are specified as follows:

- Business - for business to business (B2B) transactions. Business customers are represented as follows:
  - Buyer Organization - represents "Bill To" information
  - Receiving Node - represents "Ship To" information
- Consumer - for business to consumer (B2C) transactions. Consumer customers are represented by "Bill To" information.

Customer statuses indicate whether a customer (and this customer's contacts) can log in and place orders:

- Active - Customer and this customer's contacts can log in and place orders.
- Inactive - Customer and this customer's contacts will not be able to log in. Also, a customer will be unable to create new carts or place orders from existing carts.
- On Hold - Customer, customer contact, and this customer's orders will go on hold during order confirmation. The type of hold is configurable.

For the purpose of product and pricing entitlement, customers can be further defined by the following attributes:

- Vertical - indicates which industry a customer belongs to, such as Education, Government, and so forth.
- Relationship Type - indicates whether a customer is a Reseller, a Distributor, a Contract Manufacturer, and so forth.
- Customer Level - indicates tiered values representing a customer's status, such as Gold, Platinum, Silver, and so forth.

3.2.2.2 Vendor Definition

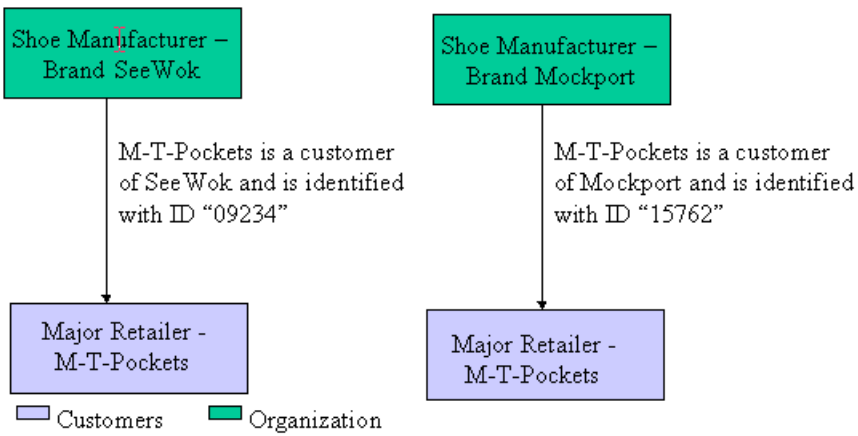
Similar to customer definition, an Enterprise can define its vendor. When defining vendors, a reference must be made to the Selling and Fulfillment Foundation organization but vendor identifier can be different from the Selling and Fulfillment Foundation organization identifier.

A vendor can be a Seller Organization, Shipping Node, or both. A vendor is represented by "Ship From" information.

3.2.2.3 Customer Modeling in a Traditional ERP System

In a traditional ERP system where a shoe manufacturing company manufactures two major brand named shoes, each shoe line is set up as a separate company (SeeWok and Mockport). Both these companies have a common customer; a major retailer called M-T-Pockets. Since these shoe lines are modeled as two different companies in the ERP system, the same customer is modeled twice without having any link to each other. In this model, all customer compliance rules (routing, labeling, payment terms, etc.) must also be defined twice for the major retailer; M-T-Pockets. Figure 3–2, "Traditional ERP System Customer Model" illustrates this scenario.

Figure 3–2 Traditional ERP System Customer Model

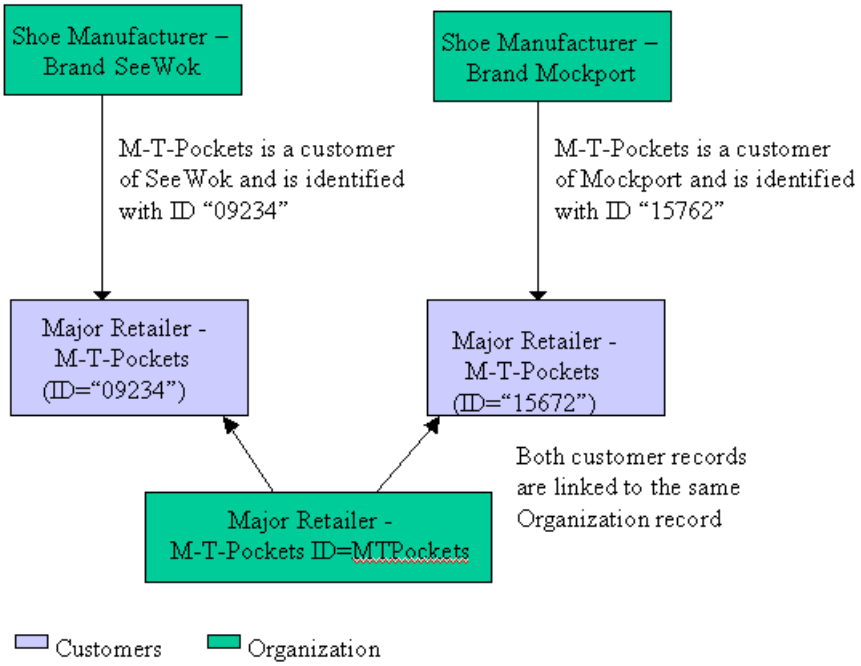


3.2.2.4 Customer Modeling in Selling and Fulfillment Foundation

Selling and Fulfillment Foundation provides two options for customer modeling in which the customer is set up as two separate customers with different Customer IDs or as two separate customers with the same Customer ID. In either case, the customer records for both customers are linked to the same organization record. Figure 3–3, "Selling and Fulfillment Foundation Customer Modeling Option 1" and Figure 3–4,

"Selling and Fulfillment Foundation Customer Modeling Option 2" illustrate these two options.

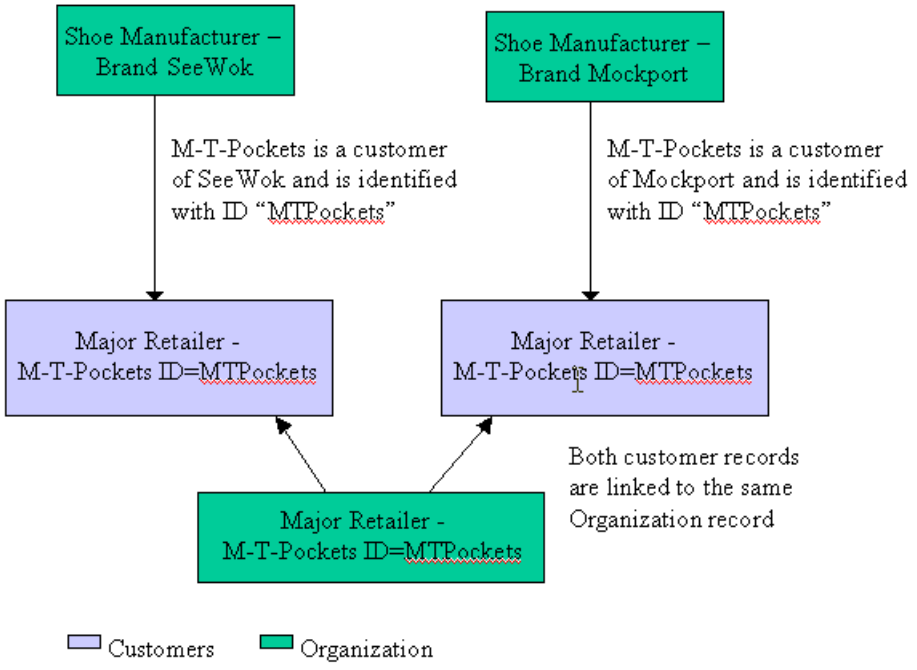
Figure 3–3 Selling and Fulfillment Foundation Customer Modeling Option 1



With this option, all customer compliance rules (routing, labeling, payment terms, etc.) defined for the major retailer must be defined only ONCE in this model for the organization "M-T-Pockets". Selling and Fulfillment Foundation automatically picks up the right configuration even though they are modeled as two different customers.

Another advantage of this approach is that this rationalization to a single organization can be done over a period of time. In the beginning, the customers could have been represented as two different organizations but over the period of time when rationalization was feasible, merged into a single entity.

Figure 3–4 Selling and Fulfillment Foundation Customer Modeling Option 2



With this option, all customer compliance rules (routing, labeling, payment terms etc.) defined for major retailer must be defined only ONCE in this model for the organization "M-T-Pockets". The system automatically picks up the right configuration even though they are modeled as two different customers.

3.2.2.5 Customer Management

In most business-to-business environments there are numerous buyers that create carts, place orders, request quotes, and so forth. Likewise, there are multiple customer service representatives (CSRs), who manage the needs of these customers and the orders that they place. However, all customer service representatives do not typically manage all customers within an organization. Each customer service representative

is usually responsible for managing the needs of a select number of customers. Customers can be grouped by a classification or by specific order attributes.

In Selling and Fulfillment Foundation, each customer service representative, or user, can be manually assigned to manage a specific customer or automatically assigned as a member of a team that is assigned to manage multiple customers. Supervisors have access to all customers who are managed by their teams, or the teams of their descendants.

Customer service representatives are user contacts who manage customer accounts for seller organizations. However, buyer users are user contacts who manage accounts for customer organizations. Buyer customer assignments allow you to assign a buyer user to one or more customer organizations. Using buyer customer assignments, customers can place their own orders with sellers, and in some cases, place orders on behalf of other customers.

Customer assignments are supported in a customer hierarchy with one or more Customer Master Organizations.

Note: To implement this functionality, you must call the updateCustomerAssignmentLookup API, either manually or through a service, whenever changes to customer hierarchy or customer assignments occur.

3.2.2.5.1 Customer Assignments

Customer service representatives (users) are assigned to customers:

- Manually
- Manually, based on Order attributes
- As a team

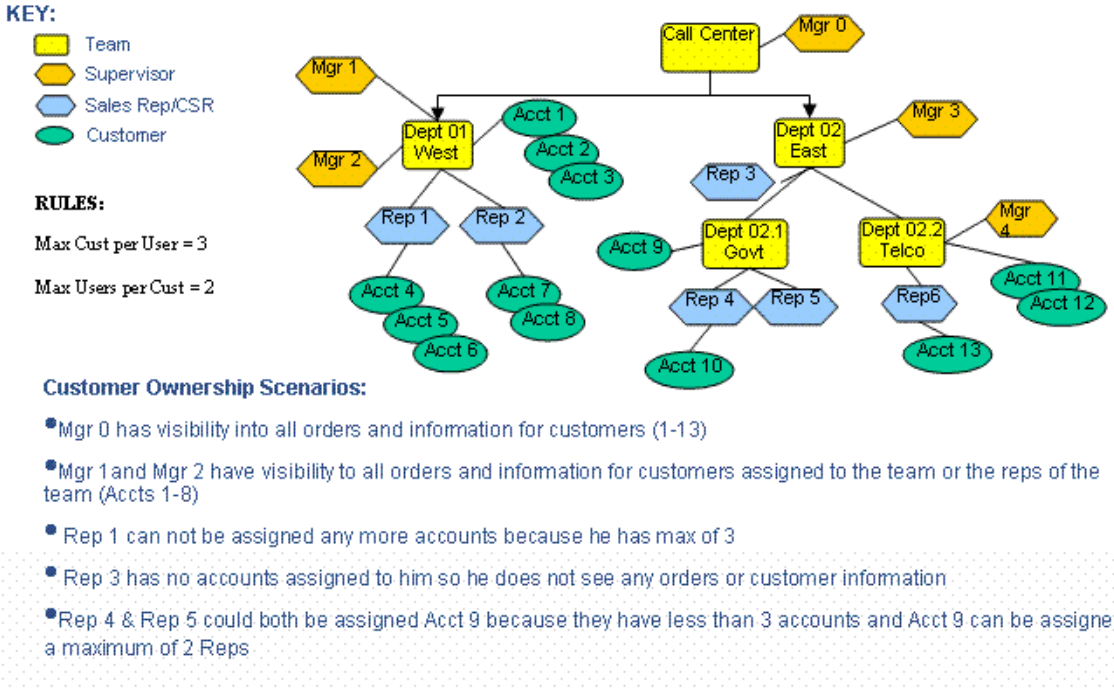
In buyer customer assignments, buyer users are assigned to a customer or a set of customers.

Manual Customer Assignment

Whenever a sales person makes the first sale to a customer, he or she can be assigned to manage that customer. In this scenario, manual assignments for each user are required so that even though two sales people belong to the same team, one cannot manage customers for the other. Whenever manual user assignments are enabled, the user must be directly assigned to the specific customer. Only then does that user have access to the customer orders and information. If a user is a supervisor, then he or she has access to orders and information for all customers that are assigned to his teams and for all customers assigned to their sub-teams.

You can define the maximum number of teams or users who can manage the customer and the maximum number of customers that can be assigned to a particular user. Figure 3–5 illustrates customer ownership scenarios for manual customer assignments.

Figure 3–5 Manual Customer Assignment



Manual Customer Assignment Based on Order Attributes

Selling and Fulfillment Foundation provides a team code attribute on orders. Users who belong to that team have visibility to those orders. External logic should be implemented to populate the team code during order creation and modification.

Customer Assignments to a Team

Often in business-to-business environments, each customer is managed by a specific team of customer service representatives (users). Selling and Fulfillment Foundation offers three customer-to-team assignment choices:

- All users on a team can access all customers assigned to their team.

  This enables all team members to see orders and receive alerts associated with the customers who are assigned to the team. To

configure this, specify that manual user-to-customer assignment is not required.

- All users on a team require direct assignment to customers.

  This enables team members to access customer orders and alerts only if they have been directly assigned to that customer. To configure this, manual user-to-customer assignment is required.

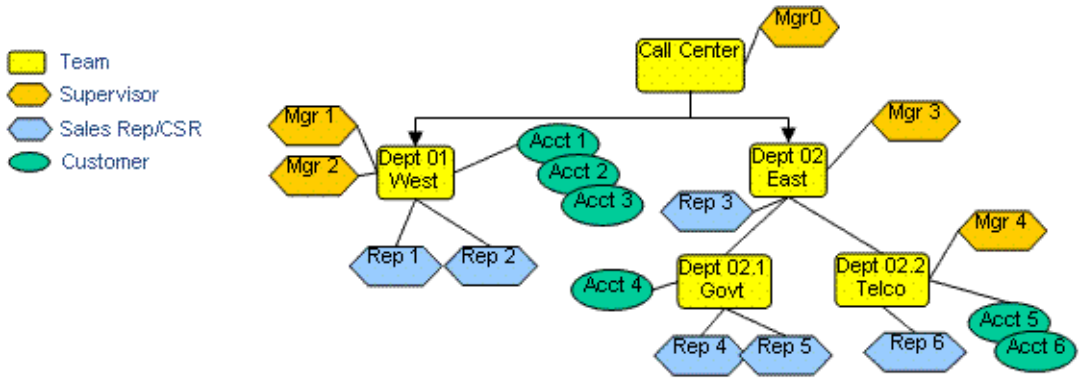
- Some teams can access all customers, while other teams require members to be assigned directly to a customer.

  Teams that can access all customers see all of their orders and associated alerts. Teams that require direct assignment to customers need access to see a customer's orders and associated alerts. In this case, manual user-to-customer assignment is required.

Team supervisors can access all orders and information for all customers that are assigned to their teams or sub-teams. Whenever a user can access a customer, he or she can access all of the descendants for this customer.

Figure 3–6 illustrates customer ownership scenarios for customer assignments to a team.

Figure 3–6 Customer Ownership Scenarios for Team Assignments



Customer Ownership Scenarios:

- Mgr 0 has visibility of orders and information for all customers (1-6)
- Mgr 1 or Mgr 2 as well as Rep 1 and Rep 2 have visibility of orders and information for all customers assigned to their team (Accts 1-3)
- Rep 3 has no customers assigned to his team so he does not see any orders or customer information
- Rep 4 & Rep 5 have visibility into Acct 4

Buyer Customer Assignments

In a business-to-business environment, multiple organizations can be modeled as customers. Customer contacts, also known as buyer users, can be assigned to manage accounts for these customers. Buyer customer assignments allow customers to manage their own accounts, and in some cases, manage the accounts of other customers. Some customers, however, do not manage their own accounts or other customer accounts.

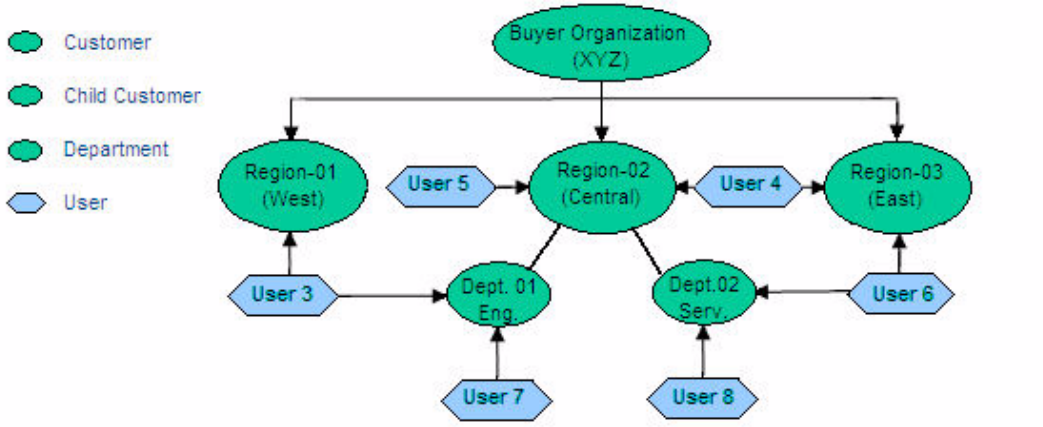
Selling and Fulfillment Foundation uses organization and user information passed through the manageCustomerAssignment API to determine buyer customer assignments. After configuring customer-buyer assignments,

data access policies for buyer users provide the following choices for account management:

- Buyer users can access data for their assigned customers only.
- Buyer users can access data for their assigned customers and child customers.

Figure 3–7 illustrates buyer customer assignments in which some customers manage accounts for other customers.

Figure 3–7 Buyer Customer Assignments



Account Ownership Scenarios:

- All users modeled here belong to XYZ Buyer Organization.
- Users 3, 4, 5, and 6 are all assigned regional offices for managing their respective supplies.
- Users 7 and 8 are assigned to specific departments for managing supplies.
- Users 3 and 6 can also be assigned specific departments to manage orders for that department.
- Users 4 and 5 can manage supplies for both departments based on hierarchy.

3.2.2.5.2 Multiple Customer Master Organizations

Selling and Fulfillment Foundation supports customer assignments across multiple customer master organizations, where users are assigned to customers from one or more customer master organizations. In this business scenario, a customer service representative or a buyer user can be assigned to customers that are not part of the same customer hierarchy.

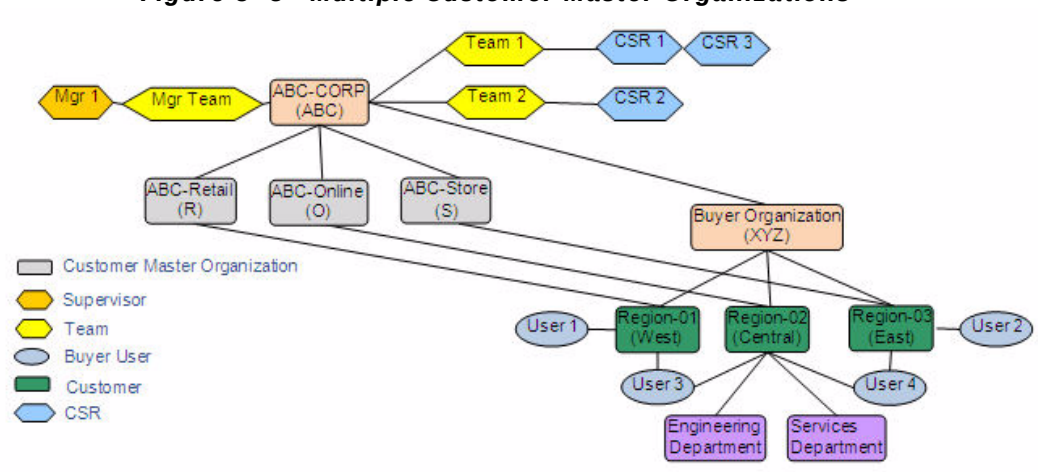
When assigning a customer service representative to customers from multiple customer master organizations, define a team for the customer service representative and then add the customer master organizations to the team's enterprise list. Next, the manageCustomerAssignment API is used to assign the customer service representative and team to customers. The customer service representative can manage orders for all the assigned customers.

When assigning a buyer user to customers from multiple customer master organizations, ensure that the customers belong to the same buyer organization. Next, the manageCustomerAssignment API is used to assign the buyer user to customers. The buyer user can manage orders for all the assigned customers.

For more information about configuring users and teams, refer to *Selling and Fulfillment Foundation: Application Platform Configuration Guide*.

Figure 3–8 illustrates customer assignments for multiple customer master organizations.

Figure 3–8 Multiple Customer Master Organizations



Account Ownership Scenarios:

- ABC-Retail, ABC-Online, and ABC-Store are customer master organizations, each having one Region as a customer.
- CSR 1's team, Team1, is assigned to the Region-01 customer and the Region-03 customer. CSR 2's team, Team 2, is assigned to the Region-02 customer. CSR 3's team, Team 1, is assigned to the Region-01 customer and the Region-03 customer.
- CSR 1 and CSR 3 can manage orders for the Region-01 customer and the Region-03 customer.
- CSR 2 can manage orders for the Region-02 customer.
- Mgr. 1 can manage all customers: Region-01, Region-02, and Region-03.
- Users of Buyer Organization XYZ (User 1, User 2, User 3, and User 4) can manage orders for their respective Regional customers.
- User 3 and User 4 can manage orders for both departments, based on hierarchy.

The example shown in Figure 3–8 demonstrates an account ownership scenario for users assigned to customers from multiple customer master organizations.

To assign CSR 1 and CSR 3 to customers from multiple customer master organizations, CSR 1 and CSR 3 are assigned to Team 1, and then the ABC-Retail and ABC-Store customer master organizations are added to Team 1's enterprise list. In this example, CSR 1 and CSR 3 can manage orders for ABC-Retail's customer, Region-01, and ABC-Store's customer, Region-03. Similarly, CSR 2 is assigned to Team 2, which has the ABC-Online customer master organization added to its enterprise list. CSR 2 can manage ABC-Online's customer, Region-02. The supervisor, Mgr. 1, can manage the orders of all customers assigned to CSR 1, CSR 2, and CSR 3.

Additionally, in the example, User 3 is assigned to customers from multiple customer master organizations. User 3 is assigned to the Region-01 customer from the ABC-Retail customer master organization and to the Region-02 customer from the ABC-Online customer master organization. Both the Region-01 customer and the Region-02 customer belong to the same buyer organization, Buyer Organization (XYZ).

3.3 Guidelines for Organization Modeling

When preparing to set up your organization hierarchical model for Selling and Fulfillment Foundation, you need to identify which roles and responsibilities you want to define for each organization.

To determine which organizations should have which roles and responsibilities assigned to them:

1. Identify each of the Organizations for your organization hierarchy. That is, business group, companies, legal entities, sales organizations and purchasing organizations.

   There can be several reasons for a business group to have multiple companies. One example is the acquisition of a company. Another example could be when the business group has two unrelated businesses.

   A company may need to have multiple legal entities if it operates in geographical areas that are different from each other in regards to legal issues such as currency, taxation structure, or other legal requirements.

   Decide the basis for defining sales organizations (for example, sales channel, product group, geography or customer). Sales organizations have sourcing rules and pricing rules. This may help you determine sales organizations. For example, if you have different pricing rules or sourcing rules for different sales channels or geographies, then your sales organizations can be based on sales channel and geography.

2. Identify Enterprises for the organizations in your organization hierarchy. Considerations for identifying Enterprises are explained through the examples in Section 3.5, "Organization Model Examples".

3. Identify Catalog organization(s). Any organization in the organization hierarchy can be the catalog organization. Typically, the main company is the catalog organization. The Catalog organization needs

to be defined at a lower level in the organization hierarchy if two organizations of the company have same item identifier for two different physical items and the company does not have a common Item ID for the two items. The Catalog organization can be defined at a lower level if two legal entities do not cross sell each other's items. But, the Catalog organization cannot be below the Inventory organization in the hierarchical structure.

4. Identify Inventory organization(s). Any organization in the organization hierarchy can be an inventory organization. Typically, an inventory organization is kept at the legal entity level. There are both advantages and disadvantages in keeping the Inventory organization at this level in the hierarchy. For example, if the Inventory organization is at the legal entity level then all sales organizations of the legal entity have equal access to the inventory in the locations of the legal entity (with the capabilities to restrict access of a sales organization to only designated physical locations within the inventory organization). This can be an advantage in one business and can be disadvantage in other business.
5. Identify physical location owners (nodes). This definition in most cases should map the real ownership of physical locations. Typically legal entities are owners of the physical asset and are modeled as owner organizations of nodes.

3.4 Installation Level Rules for Organization Modeling

Installation-level rules are used to automatically default various organizational roles for a participant.

Since organization modeling can be complex to understand, Selling and Fulfillment Foundation provides powerful defaulting rules that can be set up at the installation level.

When modeling catalog organizations, you can specify if the item master:

- Is kept at the Hub level

  You specify this if there is just one definition of the item master for the complete installation. If you choose this as the default model, all participants are automatically assigned the Hub organization as their

catalog organization at the time that the participant is created. Using an advanced setup, this default can be changed as required.

- Is kept at an Enterprise level

  You specify this if you want all participants defined as Enterprises to become their own Catalog organizations. For all other participants, the catalog organization is designated as their "primary enterprise".

- Is to be defined by every participant

When modeling inventory or capacity organizations, you can specify if:

- A single consolidated inventory or capacity view is required for the complete installation. If you choose this as the default model, all participants are automatically assigned the Hub organization as their inventory or capacity organization at the time that the participant is created. Using an advanced setup, this default can be changed as required.
- The Inventory or Capacity organization needs to be kept at an Enterprise level. If you choose this as the default model, all participants defined as Enterprises become their own Inventory or Capacity organizations. For all other participants, the inventory or capacity organization is designated as their "primary enterprise".

You can change the defaulted value of these organizations to any other organization as required using the advanced organization modeling setup. However, it is strongly recommended that you understand the implications of organization modeling before using the advanced setup provided. There are advantages and disadvantages in each approach and a good understanding of this document can help you make the right decision.

3.5 Organization Model Examples

This section provides examples of how you might model the organizations of an Electronics company and a Third-Party Logistics company.

3.5.1 Electronics Company Model

Alphabet Electronics Corporation has two companies, ABC and XYZ.

ABC has one legal entity in North America and one in the Asia Pacific. ABC has two sales organizations; one sells electronic gadgets (called, Sales-Electronics) and other sells music and movies (called, Sales-Entertainment).

ABC North America Sales-Electronics has one distribution center in California and one in Massachusetts.

ABC North America Sales-Entertainment has two distribution centers in California, one for music and the other for movies. It also has one distribution center in Massachusetts for both music and movies.

ABC North America has one purchasing organization for its distribution centers in North America.

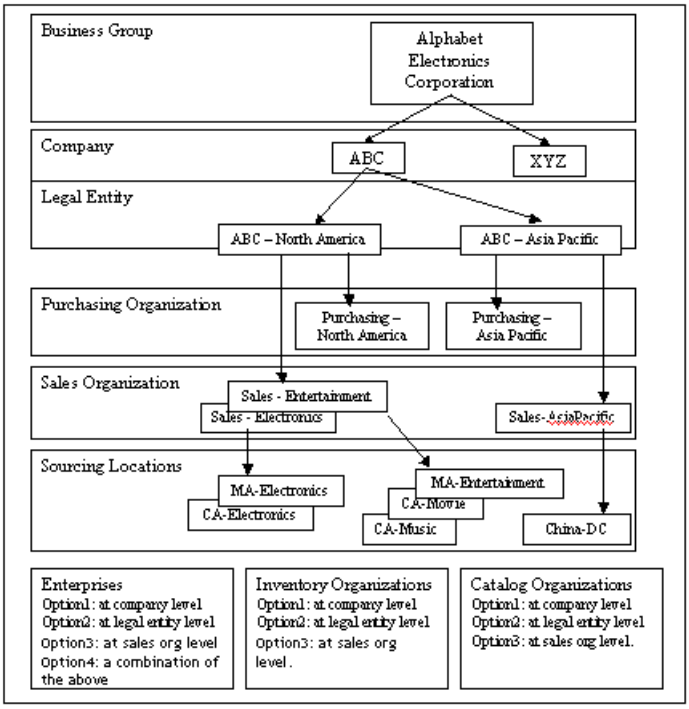
ABC Asia Pacific has one sales organization for both electronics and entertainment. It has one distribution center in China. It has one purchasing organization for the distribution centers in Asia Pacific.

The organization structure of the XYZ company is the same as ABC.

3.5.1.1 Organization Hierarchy

The organization hierarchy for the Alphabet Electronics Corporation is illustrated in Figure 3–9, "Organization Hierarchy of the Alphabet Electronics Corporation". For simplicity, this illustration shows the organization hierarchy for the ABC company entity only. The organization hierarchy for the XYZ company is the same as ABC.

Figure 3–9 Organization Hierarchy of the Alphabet Electronics Corporation



3.5.1.2 Choosing the Enterprise

The decision about the hierarchy level at which you should have Enterprises defined depends on the following:

- Who defines the business rules and process definition?

  The "Enterprise" owns and controls the document flow and business rules associated with the document flow. If the sales organization needs the ability to define their own process definition and business rules then your Enterprise needs to be at the sales organization level.

- Where does the customer service organization fit in the organization structure?

An order can be accessed only by an organization defined as an Enterprise, Buyer, or Seller organization on an order. If customer service is managed centrally across sales organizations, you may want to create the "Enterprise" at the same level as your customer service organization.

Typically Enterprises should be modeled as high as possible in the organization structure. This ensures common business practices between different groups. Note that even though Enterprise responsibility is designated at a higher level, process definition can still be differentiated for each of the sales organizations by effectively using pipeline determination rules.

Selling and Fulfillment Foundation supports the definition of multiple organizations in a single hierarchy to be designated as "Enterprises". This enables you to model the real organization structure in Selling and Fulfillment Foundation and still get the capability of managing all processes at the lowest levels in your organization hierarchy.

3.5.1.3 Choosing the Inventory Organization

The decision about the hierarchy level at which you should have your inventory organizations defined depends on the following:

- Can inventory ownership be established by virtue of physical locations?

  Your inventory organization provides a clean segregation of inventory between two inventory organizations. If physical locations can establish the legal ownership of your inventory, this feature may not be necessary to establish that ownership. If you can afford that luxury then your inventory organization can be defined up higher in the organization hierarchy allowing better and expanded inventory views across multiple organizations.

- Does an organization need the ability to be able to source its sales order from another organization?

  Currently, Selling and Fulfillment Foundation does not allow order promising and scheduling for the sales orders of one selling organization from any other inventory organization but its own (except when inventory is maintained externally). If organizations belonging to two different legal entities must be able to source from each other, then they must share a common inventory organization.

This situation can also arise when vendors publish inventory information to organizations. A common inventory organization is designated for all vendors and the organization needing access to vendor inventory. This allows organizations to have visibility to their vendor's inventory also.

Note that even though an organization can have only one inventory organization, all supply update transactions in Selling and Fulfillment Foundation provides the capability to specify the "Inventory organization" for which the transaction is being carried out. This provides a powerful feature where a vendor can publish their inventory information to multiple inventory organizations at the same time. If the inventory is published into two different inventory organizations for the same product, the vendor must ensure that they have logically segmented their inventory to avoid over allocations.

- Should I model the sales organizations or the legal entities as inventory organizations?

  Inventory organizations can be modeled at the legal entity level without much concern if:

  - Your sales organizations are involved in selling of completely different products
  - The same product is sold by multiple sales organizations but each sales organization has complete ownership of a physical location and thus inventory. Note that access still needs to be controlled through distribution groups to prevent one sales organization from accessing another sales organization's physical location.

  If the same physical location is shared by multiple sales organizations involved in selling the same product and you need clear separation of inventory for each sales organization then the inventory organization must be modeled as each sales organization. In the future, Selling and Fulfillment Foundation may provide enhanced segmentation capabilities to handle such situations.

  It is strongly suggested that you keep inventory organizations at the legal entity level, as this greatly enhances the inventory visibility across organizations.

- Should I model legal entities or the company as inventory organizations?

Keeping inventory organizations at the company level gives visibility on the company's inventory to all organizations in the company. But to be able to do so, the following criteria should be evaluated.

- Can inventory in each physical location be tied to a legal entity? If yes, than inventory ownership does not cause problems and does not provide a roadblock. If no, can Selling and Fulfillment Foundation be oblivious to inventory ownership? In some cases this may not be necessary as Selling and Fulfillment Foundation may be used as a pure fulfillment solution with inventory ownership data being maintained in another system.
- Can a common catalog organization be established between legal entities to ensure that the same item identifiers do not represent two physical products? A common catalog organization serves the function of providing the "global item id" even when two legal entities may name the same physical item differently and in some cases, two separate physical products the same.

If the above questions are answered affirmatively, the company can be modeled as the inventory organization.

Typically, the inventory organization should be modeled at the highest level possible to provide the most visibility across organizations.

3.5.1.4 Choosing the Catalog Organization

The decision about the hierarchy level at which you should have your catalog organizations defined depends on the following:

- The catalog organization needs to be at the same level as the inventory organization or above.
- As with other setups, setting up the catalog organization at a higher level causes lesser setup and facilitates the sharing of common definitions.
- When making this decision determine whether or not item identifiers can be rationalized to have unique global identifiers within the business group or some level below it. Since item identifiers are unique for a catalog organization, this is a mandatory requirement for being able to push this definition to a higher level.

3.5.2 A Third-Party Logistics Company Model

3PL is a third party logistics company. 3PL's clients are ABC and XYZ.

3PL stores inventory for ABC and XYZ in its warehouses and fulfills orders for ABC and XYZ. The businesses of ABC and XYZ are very similar and they can have the same item identifiers representing two different physical items.

3PL's revenue comes from the service fee for processing its clients' sales orders and purchase orders. 3PL sends the order processing details to its financial organization to collect the service fees from the clients.

The inventory in 3PL's warehouses is owned by the clients. 3PL has one warehouse in California and one in Massachusetts. 3PL stores products owned by both the clients (ABC and XYZ) in the same warehouse.

The organization structure of ABC and XYZ is the same as what is described in the previous example with the following differences:

- They have operations only in North America.
- They do not own a distribution center (sourcing location). Their inventory is kept in 3PL's locations.
- The two sales organizations of ABC, Electronics and Entertainment, do not want to have visibility on each other's inventory. However, this is not the case with XYZ.

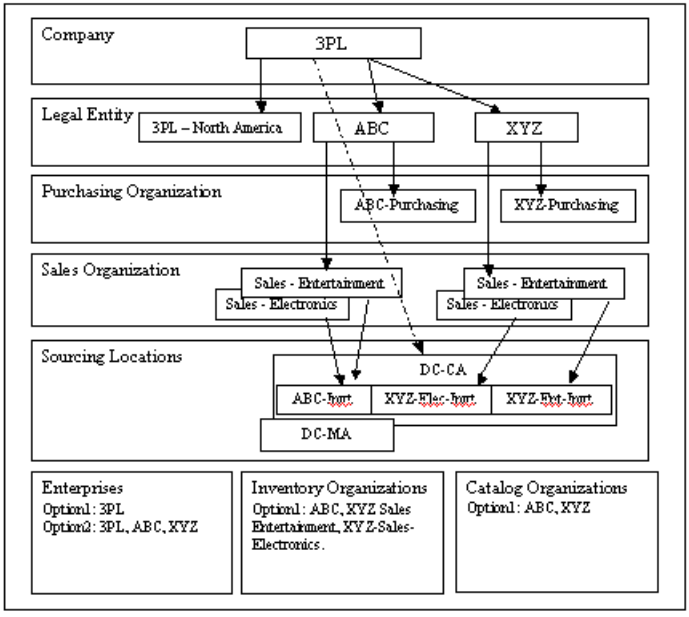
Each client generates purchase orders to replenish its stock in 3PL's warehouses. The client sends the purchase order to its vendor. It also sends a copy of the same purchase order to 3PL so that 3PL can receive the product.

Orders are managed by 3PL but the clients (ABC and XYZ) need visibility of their orders. The clients also need to have certain order modification permissions but not all. Warehouse operators also need visibility to orders and the ability to receive returns.

3.5.2.1 Organization Hierarchy

One possible organization hierarchy for the 3PL Company is illustrated in Figure 3–10, "Organization Hierarchy of the 3PL Company".

Figure 3–10 Organization Hierarchy of the 3PL Company



3.5.2.2 Choosing the Enterprise

As described in the previous example, this depends at what level you want to control the business rule definition, order fulfillment process definition and user privileges.

3PL defines and maintains business rules and order fulfillment processes for orders of its clients. So, defining only 3PL as the Enterprise can be the desirable option. But, ABC and XYZ need visibility and certain order modification permissions on their orders. Also, ABC and XYZ may want to create and maintain their users. This means, ABC and XYZ also need certain enterprise-level capabilities. So, defining 3PL, ABC, and XYZ all as enterprises is the more appropriate option. But with this option, 3PL needs to restrict permissions for ABC and XYZ so that they do not have the ability to define and maintain order fulfillment processes.

If ABC or XYZ does not want certain users (for example, customer service representatives) of its one sales organization to have the visibility

of orders of its other sales organization, then each sales organization can also be assigned the Enterprise role with restricted privileges.

3.5.2.3 Choosing the Inventory Organization

As discussed in the previous example, the inventory organizations in this model can be ABC, XYZ Sales-Electronics and XYZ Sales-Entertainment. This is because XYZ does not want its sales organization to see inventory of the other sales organization.

In situations such as third-party logistics where the same physical location is shared across multiple clients, each client should be designated as the "Inventory organization". If 3PL is dealing with multiple organizations of the same client, it can make choices of inventory organizations based on segregation level mandated by the client or client organizations.

3.5.2.4 Choosing the Catalog Organization

The catalog organization can be defined at the same level as the inventory organization or above. Since, the two sales organizations of XYZ do not have same item identifiers for two different items, the catalog organizations can be ABC and XYZ.

4

Process Modeling Concepts

Process modeling in Selling and Fulfillment Foundation enables you to set up your business workflow of orders, inventory changes, returns, payment authorizations, or many other system events.

A typical business process model consists of:

- Base Document Types and Document Types
- Process Type Pipelines
- Repositories
- Transactions
- Conditions
- Actions
- Services
- Process Modeling Tasks

4.1 Base Document Types and Document Types

Selling and Fulfillment Foundation uses base document types and document types to carry information through a configured workflow process. A base document type defines the business documents that Selling and Fulfillment Foundation handles and defines a common storage structure for all derived document types.

The following base document types are defined in Selling and Fulfillment Foundation:

- Order
- Load
- General
- Count
- Container
- Outbound Picking
- Work Order
- Opportunity

Document types are specific business documents that are derived from a base document type. For example, document types such as Sales Order and Purchase Order can be derived from the Order base document type.

The following document types are defined in Selling and Fulfillment Foundation:

- Planned Order
- Sales Order
- Purchase Order
- Return
- Template Order
- Transfer Order
- Master Order
- Load
- General
- Count
- Container
- Outbound Picking
- Work Order
- Quote

Business rules such as payment collection rules and modification rules must be set up for each document type. For more information about setting up business rules for document types, see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide*.

4.2 Process Type Pipelines

In Selling and Fulfillment Foundation, a business document, such as an Order, goes through a series of defined processes. These processes are called *base process types*. Every type of base document has a defined set of base process types. For example, the Order base document type has the following base process types defined for it:

- Fulfillment
- Negotiation
- Delivery

The Load base document type has the Load base process type defined for it.

The General base document type has the General base process type defined for it.

You configure the flow of these processes in the Process Modeling by creating process type pipelines. A ***process type pipeline*** is a series of transactions and statuses that guide document types, such as Sales Order and Purchase Order, through a related process. A pipeline consists of the different statuses a document goes through during fulfillment. You can also set up transactions consisting of events, actions, and conditions, as they pertain to the pipeline you are configuring.

The following tables detail each base process type, the process types that are derived from them, and the pipelines associated with the process type.

Table 4–1 Order Fulfillment Base Process Type

Derived Process Types	Process Type Pipelines
Order Fulfillment	Sales Order Fulfillment
Planned Order Execution	Planned Order Execution
Template Order	None

Table 4–1 Order Fulfillment Base Process Type

Derived Process Types	Process Type Pipelines
Reverse Logistics	Reverse Logistics, Consumer Returns
Purchase Order Execution	Purchase Order Execution, Drop Ship Purchase Order Execution
Transfer Order Execution	Transfer Order Execution
Quote Fulfillment	Quote Fulfillment

Table 4–2 Order Negotiation Base Process Type

Derived Process Types	Process Type Pipelines
Master Order Fulfillment	Master Order Fulfillment
Order Negotiation	Order Negotiation
Planned Order Negotiation	Planned Order Negotiation
Purchase Order Negotiation	Purchase Order Negotiation

Table 4–3 Receipt Base Process Type

Derived Process Types	Process Type Pipelines
Return Receipt	Return Receipt
Purchase Order Receipt	Purchase Order Receipt
Transfer Order Receipt	Transfer Order Receipt
Sales Order Receipt	Sales Order Receipt

Table 4–4 Order Delivery Base Process Type

Derived Process Types	Process Type Pipelines
Outbound Shipment	Outbound Shipment
Inbound Shipment	Inbound Shipment

Table 4–5 Load Delivery Base Process Type

Derived Process Types	Process Type Pipelines
Load Execution	Load Execution

Table 4–6 General Base Process Type

Derived Process Types	Process Type Pipelines
General	None

Table 4–7 'Pack Process' Process Type

Derived Process Types	Process Type Pipeline
Pack Process	Pack Process

Table 4–8 Count Execution Process Type

Derived Process Types	Process Type Pipelines
Count Execution	Count Execution

Table 4–9 Outbound Picking Process Type

Derived Process Types	Process Type Pipelines
Outbound Picking	Standard Pick Process

Table 4–10 VAS Process Type

Derived Process Types	Process Type Pipelines
VAS	VAS Work Order

Table 4–11 Opportunity Base Process Type

Derived Process Types	Process Type Pipelines
Opportunity Fulfillment	Opportunity Fulfillment

4.2.1 Pipeline Determination

Pipeline determination is used to set up conditions that affect which pipeline is used during the workflow. For example, an organization deals with sales orders that sometimes contain hazardous materials. They have two separate pipelines, one in which orders with order lines without any hazardous materials go through the normal order process and one in which orders with order lines containing hazardous materials must go through inspection before continuing through the order process. The

organization uses pipeline determination to set up a condition that determines whether or not order lines contain hazardous materials and sends the order down the correct pipeline.

4.3 Repositories

A *repository* is a logical collection of entities that define a given business process.

The following entities are included in a repository:

- Pipelines
- Transactions
- Statuses
- Conditions
- Actions
- Services

Selling and Fulfillment Foundation provides a repository for each of the process types. When creating a new process type from a base process type, the corresponding base repository entities are copied and attached to the new process type. For example, when a Sales Order Fulfillment process type is created from the Fulfillment base process type, the base repository entities contained in Fulfillment are copied and attached to Sales Order Fulfillment.

4.4 Transactions

Every base process type has a set of base transactions defined for it. A *transaction* is a logical unit of work that is necessary for performing activity within Selling and Fulfillment Foundation. Base transactions are predefined transactions that contain information about how the transaction behaves, such as how many copies of a transaction can be kept in a process type and whether or not it can have configurable base pick and drop statuses. Base transactions can be used to create new transactions. These transactions can be changed within the limits defined in the base transaction.

In Selling and Fulfillment Foundation, APIs are used to complete transactions. When an API is invoked, the Transaction ID is determined

based on the context that the API was completed. The *transaction ID* identifies the transaction to be completed. Depending on the situation, the transaction ID can be passed as an input parameter or it can be pre-defined for the invoking API. For more information about APIs, see the *Selling and Fulfillment Foundation: Javadocs*.

Transactions can be classified as one or more of the following types:

- Externally-triggered
- User-triggered
- Time-triggered

Externally-Triggered Transactions

An *externally-triggered transaction* is performed through the Selling and Fulfillment Foundation Services Definition Framework which calls a corresponding API within Selling and Fulfillment Foundation to complete the transaction.

User-Triggered Transactions

A *user-triggered transaction* is performed based on user actions performed in the Selling and Fulfillment Foundation User Interface, configured alert queue, or an e-mail exchange.

Time-Triggered Transactions

A *time-triggered transaction* is performed on scheduled intervals. In Selling and Fulfillment Foundation, a time-triggered transaction is also called an agent.

4.4.1 Transaction Dependency

In the order fulfillment cycle, you can have specific products and services in an order that need to be fulfilled in a particular sequence. You can set up dependencies on a transaction such that an order line cannot be processed until certain conditions are met. The dependencies enable you to define rules based on item classifications, item IDs, service types, and many other conditions that can be defined for on an order line.

Transaction Dependencies can be configured at a specific enterprise, document type, and process type level. For example, a dependency

specified by an enterprise for a schedule transaction applies to all the sales orders of that enterprise, regardless of the pipeline used.

Before processing, each order line is checked for any dependencies on the transaction. If multiple dependencies are defined for a transaction, all the dependencies have to be satisfied before the line can be processed.

Dependencies can also be defined for bundle components and based on any order date. In the case of the bundle components, the order lines may have interdependencies while fulfilling a bundle order. The bundle parent line cannot be invoiced until every child line has completed the specified transaction. Therefore, it is necessary to allow for transactions to understand that an order line cannot be processed until certain conditions are met. For more information about bundle components, see Section 7.5, "Bundles".

The order date related dependency can be configured for a transaction based on specific order dates. You can specify these dependencies using templates provided with Selling and Fulfillment Foundation. For more information about configuring this dependency, see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide*.

Transaction Dependency Usage Scenario

The transaction dependency for an order can be used in the following scenario:

For any order line with item classifications of MODEM, FILTER, or CABLE, the schedule transaction is not allowed to process until 24 hours after all the order lines containing items with the classification of LINE ACTIVATION have completed the ship order transaction.

If this dependency is specified, the schedule transaction first checks to see if there is an item with the MODEM classification in the order. If the item is identified, other lines on the order are checked to see if any of them have a LINE ACTIVIATION class item. For each line with an item classification of LINE ACTIVATION, it is then determined if it has been at least 24 hours since that line reached the “SHIP_ORDER” transaction.

Transaction Dependency Halting Other Transactions

When a transaction dependency halts an order-level transaction, such as payment processing, a flag is returned in the output XML of the API to indicate that the order cannot be processed due to dependencies. The flag is not returned in the output XML of all APIs. For more information,

see the *Selling and Fulfillment Foundation: Javadocs*. Since most transactions are status-based, they ignore any lines that have not met the dependencies. However, when an agent encounters an order that cannot be processed, either in whole or in part due to a dependency, it silently ignores whatever lines cannot be processed and updates the task queue to process them at a later time.

Overriding Dependencies

To override transaction dependencies, you can pass a flag to the API that invokes the transaction so that the transaction dependencies are ignored. This flag is only supported by some APIs. For more information about APIs, see the *Selling and Fulfillment Foundation: Javadocs*.

Circular Dependencies

Circular dependencies arise when a transaction is dependent on another transaction that is waiting for the main transaction to either change its status or complete. You can set up dependencies in such a way that a transaction may never process an order due to a circular dependency.

For example, there may be a rule that says ITEM1 cannot be scheduled until ITEM2 has completed the schedule transaction. There could be another rule that prevents ITEM2 from scheduling until ITEM1 has scheduled. As a result, a circular dependency arises. In this case, the circular dependency is detected by the transaction dependency logic and an exception is thrown along with a list of the dependencies that caused this circular loop.

4.4.1.1 Supported Transactions

The system transactions that support dependencies are identified by a flag in its base process type.

This flag cannot be edited for system or built-in transactions. You can enable or disable the dependencies only for extended or custom transactions. Keep in mind, that system and custom listeners do not support transaction dependency. However, you can configure transaction completion for each instance of the listener. For more information about configuring transaction completion, see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide*.

The supported transactions are:

- Chained Order Create - This transaction ignores the order lines that do not meet the dependencies.
- Close Order - This agent ignores the order if any of the order lines do not meet the dependencies.
- Payment Collection - This API returns a flag to indicate that the dependencies are preventing the transaction from processing if any line has unmet dependencies. However, the agent ignores the current order and processes it in the next run.
- Payment Execution - This API returns a flag to indicate that dependencies are preventing the transaction from processing if any line has unmet dependencies. However, the agent ignores the current order and processes it in the next run.
- RELEASE - This transaction ignores the lines that do not meet the criteria and populates the schedule failure reason for any line that was not processed due to the dependencies.
- SCHEDULE - This transaction ignores the lines that do not meet the criteria, and populates the schedule failure reason for any line that was not processed due to the dependencies.

Abstract transactions have a different flag to indicate the type of dependency supported.

The two types of dependencies that are supported by abstract transactions are:

- None - The support dependency for an individual instance of the transaction is always N.
- Instance Specific - The support dependency is configurable for each instance of the transaction.

By default, all abstract transactions have this dependency flag set to None. However, the following abstract transactions support specific instances such that they ignore any line that does not meet the dependencies:

- Change Order Status
- Create Order Invoice
- Derived Order Create

- Send Order.

For more information about configuring transaction dependencies, associated rules, and constraints, see the *Sterling Distributed Order Management: Configuration Guide*.

4.4.1.2 Task Queue Updates

When a task queue-based agent is prevented from processing a transaction with dependencies, the date when the order is ready to process is calculated.

Upon completing the transaction, the dependencies are examined to see if there are any lines waiting on the completion of this transaction. The task queue dates for dependent transactions are then re-calculated. Additionally, when a change is made to the order date, ship date, or delivery date, dependencies are re-evaluated similarly.

An example usage scenario for the next available date calculation is discussed in "Next Available Date Calculation".

The agents for the transactions are updated to apply transaction dependencies to the order lines before processing. If the order lines are not processed due to dependencies, the agent updates the task queue date.

Next Available Date Calculation

The next available date is calculated to identify when the order is ready to be processed. The following example provides the method of calculation for the next available date.

For example, an order has two order lines. The second order line cannot be scheduled until 24 hours after the first order line has completed the SHIP_ORDER transaction.

Until the SHIP_ORDER transaction on the first order line is completed, the earliest available date for scheduling that order line is set to:

sysdate + 24h.

When the SHIP_ORDER transaction on the first order line completes, the task queue date for scheduling the second order line is set to:

sysdate + 24h.

The next task queue date for the entire order is determined by computing the earliest available date on each of the remaining dependencies. The lower boundary of each line's maximum is the date, which is set. For example:

- Line 1 – dependency 1 - earliest date = sysdate + 5h
- Line 1 – dependency 2 - earliest date = sysdate + 48h
- Line 2 – dependency 1 - earliest date = sysdate + 12h

The earliest date for Line 1 is the maximum of (sysdate + 5h) and (sysdate + 48h) or sysdate + 48h. However, for the entire order the minimum across the lines is calculated as:

Line 2's sysdate + 12h.

Finally, if the transaction had pushed the task queue date out, the minimum of this date and the task queue date of the transaction is used. For example, if one line cannot be scheduled because of a dependency, and the other line is backordered. If the backorder relog interval is less than the calculated time for the remaining dependencies, the backorder relog interval is used instead.

4.4.1.3 Transaction Completion

The transactions configured in the pipeline changes the status of the order line after successful execution. A transaction is either complete or incomplete for an order line, based on the status change. All the transactions need not understand the completion. This is based on the configuration. The completion is recorded only if that transaction can be configured for completion. Once the transaction is enabled for completion, it is evaluated to be marked as completed or incomplete whenever any status change happens to the order line. In some cases, a completed transaction can be marked as incomplete since its status could be demoted.

There are two ways to mark a transaction as complete or incomplete. One way provides a single status and whenever the order line goes past, this status transaction is marked as complete. The other way to mark a transaction as complete is by providing a list of statuses. If the order line is in any of those statuses, the transaction is marked as complete. To avoid configuring a list of statuses, it is better to configure statuses in an ascending order as the order's life cycle progresses.

For example, if transaction X needs to be completed before transaction Y, then transaction X should drop in status which is less than the status transaction Y drops in to.

The order fulfillment pipeline has transactions pre-configured for completion. You cannot modify the completion criteria for system transactions. However, you can configure the completion for custom or derived transactions. You can also configure completion for derived listeners, but it has to be done for every instance of the listener.

An event is triggered when a transaction is completed and it is determined that the order is ready to process the dependent transaction. This event also allows the custom transactions to know that the order is ready for processing.

There can be multiple transactions that become ready due to one transaction completion. The event is raised for each transaction on the order line that were waiting for the completion of this order line.

For more information about transaction completion configuration and the raised event, see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide*.

4.4.2 Events

An ***event*** is a specific occurrence in the business process; often a status change or generated alert. Releasing an order and cancelling an order are both examples of events. When an event occurs in a transaction an action is triggered.

4.4.3 Statuses

Statuses are the actual states that a document changes to and from as it moves through the pipeline. A transaction can contain two types of statuses, a drop status and a pickup status. A document is moved into a **drop status** when a transaction and its events have been completed. A **pickup status** takes the document from the drop status and moves it through the next transaction. "Created" and "Scheduled" are examples of statuses.

4.5 Conditions

A ***condition*** matches document type attributes against decision points and routes the document to the appropriate path based on the specified attribute and value combinations. The document type attributes against which conditions can be created are pre-defined in Selling and Fulfillment Foundation. You can use these attributes in any combination or you can create conditions that run the appropriate application logic for specific circumstances.

For example, at a certain point in a Sales Order Fulfillment process-type pipeline you set up a condition to determine if an order contains hazardous materials. When an order reaches this condition in the pipeline, it cannot move any further until the condition is met with a definitive Yes or No value. In this example, if the order contains no hazardous materials, the value is No and the order continues through the regular pipeline. If the order does contain hazardous material, the value is Yes and the order is sent down an alternate branch of the order pipeline that has been configured to deal with hazardous material orders.

4.5.1 Advanced XML Conditions

An advanced XML condition provides for a declarative mechanism to operate on an input XML document and evaluate conditions using the XPath expressions. These advanced XML conditions are also referred to as Greex Rules. The syntax of the advanced XML condition is XML based.

An advanced XML condition helps you to define XML based conditions or rules based on Greex syntax. Some of the most significant features of advanced XML conditions are:

- XML awareness
- Support for namespaces
- Support for decision tables
- A set of built-in libraries.
- Ability to return an XML element, a String, or Boolean.

The advanced XML condition constructs are capable of being either nested by using multiple IF and ELSE blocks or grouped by using either an AND or an OR operator. Each expression comprises one or more

function calls. These functions can be nested, meaning parameters to functions can be other function calls.

A set of these functions is provided as part of the Greex library. These functions can be called on input data via XPath expressions. Constants can be used in function parameters as well. The Greex library is available for use as a jar. It contains no dependency on either the existing platform or other components. Therefore, it is to be used as a third party library.

Table 4–12 explains function provided by the Greex Framework as part of the Greex library.

Table 4–12 Greex Framework Functions

Functions	Description
intSum	This function takes the XPath of an attribute of type Integer as argument. The function calculates and returns the sum of the values of the specified attribute.
IsTrue	This function takes the XPath of an attribute as argument. The function returns true if the value of the attribute evaluates to true.
dateGreater	This function takes two Date objects as arguments. The function returns true if the first date is greater than the second date.
IntGreater	This function takes two Integer objects as arguments. The function returns true if the first value is greater than the second value.
stringBegins	This function takes two String objects as arguments. The function returns true If the first string begins with the second string.
dateMin	This function takes the XPath of an attribute of type Date as argument. The function compares the values of the attribute and returns the earliest date.
doubleSum	This function takes the XPath of an attribute of type Double as argument. The function calculates and returns the sum of the values of the specified attribute.

Table 4–12 Greex Framework Functions

Functions	Description
isVoid	This function takes the XPath of an attribute as argument. The function returns true if the value of the specified attribute is null or an empty string.
doubleGreater	This function takes two Double objects as arguments. The function returns true if the first value is greater than the second value.
equals	This function takes two Objects as arguments. The function returns true if the first Object is equal to the second Object.
count	This function takes the XPath of an attribute as argument. The function calculates and returns the count of the values of the specified attribute.
dateAdd	This function takes two Date objects as arguments. The function calculates and returns the sum of the values of the specified attribute.
equalsIgnoreCase	This function takes the XPath of an attribute as argument. The function returns true if the first value is equal to the second value. During comparison, it ignores the case.

The Greex Framework supports the following type of advanced XML conditions or Greex rules:

- Normal Advanced XML Conditions—In this type of advanced XML condition, you can specify nested condition criteria using multiple IF and ELSE blocks. A normal advanced XML condition can return an XML element, a String, or a Boolean value.

  These type of advanced XML conditions are useful in cases where you need to define multiple condition criteria for an advanced XML condition you need to define multiple condition criteria and each condition criteria has different attributes associated with it.

For example, you may want to create an advanced XML condition for the following condition criteria:

If Ordertype="WEB" and OrderQty="100", then Discount="20".

Else

If OriginalTotalAmout="1000" and OrderLine>"5", then

TaxExemptFlag="Y".

Else HoldFlag="True".

In this case, we have multiple condition criteria and each condition criteria has different attributes associated with it. The first condition criteria has Ordertype, OrderQty, and Discount associated with it whereas the second condition criteria has OriginalTotalAmount, OrderLine, and TaxExemptFlag associated with it.

- Decision Table Based Advanced XML Conditions—In this type of advanced XML condition, you can specify nested condition criteria in a single advanced XML condition. You can have just one advanced XML condition with an array of constant values instead of one constant value. Also, in a decision table based advanced XML condition you need only one condition, hence you have only one IF block and no ELSE block. A decision table based advanced XML condition can only return a String.

  These type of advanced XML conditions are useful for cases in which you have multiple nested condition criteria to be defined for an advanced XML condition and each condition criteria has same attributes associated with it. In such cases, you can write just one condition and have a table of parameters that works like a switch statement.

  For example, you may want to create an advanced XML condition for the condition criteria, which contains different cases for the same type of attributes.

  If Ordertype="WEB" and OrderLineQty="200", then Discount="5".

  Else

  If Ordertype="STORE" and OrderLineQty="500", then Discount="7"

  Else

  If Ordertype="CALL" and OrderLineQty="250", then Discount="3".

Else

default="0"

Table 4–13 describes the above scenario in the form of a decision table.

Table 4–13 Decision Table

Order Type	Order Line Quantity	Discount
WEB	200	5
STORE	500	7
CALLCENTER	250	3
default 0		

In this case, we have multiple condition criteria but each condition criteria has same attributes associated with it. All the condition cases have Ordertype, OrderLineQty, and Discount attributes associated with it. If none of the condition cases defined are satisfied, the Greex engine returns the default value.

An advanced XML condition or Greex rule can only be created through the IDE tool called the Sterling Greex Editor. An advanced XML condition is defined in the advanced XML file also known as Greex file. For more information about creating an advanced XML condition using the Sterling Greex Editor, see the *Selling and Fulfillment Foundation: Extending the Condition Builder Guide* .

You are not allowed to modify the structure of an existing advanced XML condition or Greex rule through the Applications Manager. You can only assign new values to modifiable parameters of an advanced XML condition. For more information about modifying an advanced XML condition, see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide*.

You can localize an advanced XML condition or Greex rule by localizing the Greex file. The Greex file can be localized by implementing BundleReslover interface. For more information about localizing Greex file, see the *Selling and Fulfillment Foundation: Localization Guide*.

You can also log information about an advanced XML condition or Greex rule at different levels by implementing GreexLogger interface. For more

information about logging information about an advanced XML condition or Greex Rule, see the *Selling and Fulfillment Foundation: Extending the Condition Builder Guide* .

4.6 Actions

An ***action*** is a process or program that is triggered by an event. These processes and programs send alert notifications and automatically resolve issues.

For example, when an order is released (the event), you can set an action to send the customer an e-mail message.

4.7 Services

Services define the business process flow between Selling and Fulfillment Foundation and external systems.

4.8 Process Modeling Tasks

The tasks necessary to complete the Process Modeling include:

- Loading process-type repositories
- Creating and modifying a pipeline
- Creating, modifying, and deleting transactions
- Adding an event to a transaction
- Modifying and deleting an event of a transaction
- Adding a pickup status to a transaction
- Deleting a pickup status from a transaction
- Adding a drop status to a transaction
- Deleting a drop status from a transaction
- Setting up event handling
- Creating, modifying, and deleting a status
- Setting up a status monitoring rule definition
- Creating, modifying, and deleting a condition

- Viewing all entities affected by a condition
- Creating, modifying, and deleting an action

5

Catalog Management

The information in this chapter has been moved to the Catalog Management Concepts online file.

6

Global Inventory Visibility

Global Inventory Visibility involves acquiring and monitoring inventory levels so the product is available for shipment when a customer wants it.

6.1 Inventory Identification

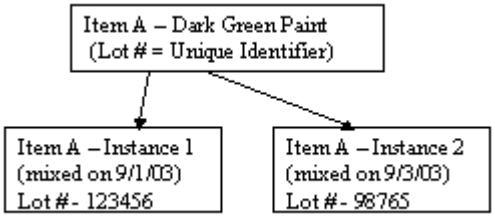
Selling and Fulfillment Foundation uses inventory identification numbers to differentiate products. This is also the case for products that require differentiation both physically and systematically for "product instances" that have slightly different characteristics. Some common examples of such identification numbers are lot number, revision number and manufacturing batch number or manufacturing dates. Different industries may have their own identification numbers. For example, some companies dealing with steel identify a roll of steel with a "mill certificate number".

These identification numbers are not necessarily relevant to every product a company sells. For some products a lot number uniquely defines all characteristics of a product whereas a revision number differentiates another product. Selling and Fulfillment Foundation uses the term "Inventory Tag Number" to rationalize these unique product identification situations. In most cases it is expected that the Inventory Tag Number represents one of the real life manufacturing or customization numbers such as lot number, batch number, or revision number based on the product. With small product extensions, the Inventory Tag Number can also represent a combination of identification numbers for cases where two inventory identification numbers together uniquely identify the product.

In most cases, the Inventory Tag Number is used by the Sterling Global Inventory Visibility and the physical product carries the actual relevant inventory identification numbers.

Figure 6–1, "Lot Number used as Inventory Tag Number" illustrates a paint product that uses a lot number to identify the specific blend that was mixed at the same time. In this case, the inventory tag number is the same as the lot number.

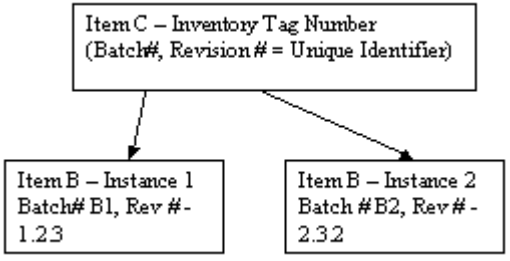
Figure 6–1 Lot Number used as Inventory Tag Number



In cases where the Inventory Tag Number is different from the product identification or is a combination of product identifiers, Selling and Fulfillment Foundation stores both the identifiers for the item.

Figure 6–2, "Combination Batch and Revision Tag Number"illustrates a product that uses a batch number and revision number combination to uniquely identify the product.

Figure 6–2 Combination Batch and Revision Tag Number



In this case, Selling and Fulfillment Foundation stores both the revision # and the batch# for the inventory item. The getTagNo() user exit is provided to generate the Inventory Tag Number and inquire on it whenever necessary. This user exit can have its own logic of merging the two identifiers or storing the combination in a separate table and returning an external tag number. This external tag number can be used as the unique inventory identifier in Selling and Fulfillment Foundation.

The Selling and Fulfillment Foundation stores tag information and has the ability to track tag numbers and ship by dates for all shipments. However, if a node does not use the Selling and Fulfillment Foundation software and cannot track tag information, it cannot inform Selling and Fulfillment Foundation about these details of a product that has shipped. For such nodes, tag tracking is not supported.

Note: In Selling and Fulfillment Foundation, tag number and ship by date are completely unrelated. The assignment of a lot number to an item does not cause the assignment of a ship by date to the item.

6.2 Supply and Demand

In Selling and Fulfillment Foundation, the supply for an item is the entire quantity of the item received at a node. Supply includes the on-hand supply. The supply consists of purchase orders (POs) and advance shipment notices (ASNs) received by the node. In addition to on-hand supply, supply includes future inventory—previously placed purchase orders that have not arrived from vendors.

In Selling and Fulfillment Foundation, the demand for an item is the expressed desire to consume a quantity of the item. Demand includes orders placed and reservations made for an item.

Demand can only be fulfilled if sufficient supply exists. The quantity available is the amount of supply left over after all demands for the item are fulfilled. Therefore, the available inventory is the difference between its supply and its demand.

6.2.1 Reservations

In Selling and Fulfillment Foundation, demand is composed of different types of specific entities called “demands.” One type of demand is a

reservation. How do you guarantee a customer that inventory has been put aside for them? One way is to create a reservation on the system, which the customer must confirm or cancel. A reservation is a quantity of an item that the seller puts aside for a customer who has the intent to purchase the items at a later date. This takes an amount of inventory out of available inventory to cater to a customer's specific demand. A reservation can be upgraded to an order or it can be canceled.

For example, you manufacture snow shovels. Your customer, the ABC Hardware store, "reserves" 150 snow shovels for November 5th. You create an order for 150 snow shovels in "reserved" status. The ABC Hardware store can either cancel the reservation or have the snow shovels shipped.

A reservation can be scheduled to expire by passing a reservation expiration date or time. The expiration date can be specified in an individual order, or a default expiration time can be configured, after which existing reservations are cancelled. This feature allows you to clean up reservations that may not have been cancelled for any reason.

Note: You cannot reserve inventory that has been segmented. For more information about segmented inventory, see Section 6.2.2, "Segmentation".

6.2.2 Segmentation

In Selling and Fulfillment Foundation, supply is composed of different types of specific entities. A type of "supply" is inventory segmentation. How do you guarantee a customer that inventory has been put aside for them? Inventory segmentation is the apportion of inventory into segments. An inventory segment is a certain amount of inventory set aside to cater to the demand from a group of privileged customers. This inventory is what is required to fulfill commitments and contracts. When inventory is apportioned for a segment, it indicates that the inventory is not to be consumed for demands other than the demands with matching segments segment types, or both.

For example, you are a shampoo manufacturer or supplier. You have contracts with a major pharmaceutical chain and a department store chain to supply them each with one of your brands of shampoo according to their specifications. When they place an order, their demand is recorded with the pre-arranged segment or segment type. The demand

is fulfilled from the supply with the matching segment or segment type at the node for the shampoo manufacturer or supplier.

> **Note:** You cannot reserve inventory that has been segmented. For more information about reserved inventory, see Section 6.2.1, "Reservations".

6.2.3 Inventory Availability Monitoring

E-commerce businesses often need real-time inventory availability indicators so that they can provide to their customers snapshots of the inventory picture without constantly making calls to Selling and Fulfillment Foundation. This can be very helpful on web sites where orders are placed, and the inventory is being viewed and modified all the time. Examples of inventory availability indicators are In Stock, Low, Limited, and Backorder/Pre-order, and Out of Stock. The criteria for each indicator level can be defined in the Applications Manager.

Selling and Fulfillment Foundation provides this functionality through the real-time inventory availability monitor. There are three ways in which it can run:

Activity Based Mode

In this mode, Selling and Fulfillment Foundation keeps track of inventory changes in real time. If the inventory level of a given item goes above or below a threshold defined in the monitoring rule of the Applications Manager, Selling and Fulfillment Foundation publishes the updated inventory level to external systems.

Quick Sync Mode

When running in this mode, Selling and Fulfillment Foundation sends the most recent inventory availability information recorded by the monitor out to external systems. If an item's inventory level went from In-Stock to Low and then back to In-Stock, Selling and Fulfillment Foundation only publishes the 'In-Stock' level for that item at synchronization time.

Additionally, since inventory availability information includes on-hand and future availability, this mode can be used to send availability messages to the planning and promotion systems.

Full Sync Mode

Typically, an enterprise runs this as a scheduled job, generally at night time. Inventory availability information is sent for all items, regardless of whether or not they have been through availability changes.

The Full Sync mode is expected to be used the first time the inventory availability monitor is run, if inventory information has not been loaded into Selling and Fulfillment Foundation through the Selling and Fulfillment Foundation APIs or Services.

> **Note:** Even when running in activity-based mode, the inventory availability monitor does not completely operate in real-time. Inventory changes need to be published to a database table, which needs to be processed by the monitor. You should therefore think of it as near real-time.

6.3 Optimization

Inventory optimization occurs when you fine tune the amount of available inventory according to the orders you expect to receive. The concept of “lead + processing time” helps you to optimize inventory.

Consider an item with 20 units available in the warehouse ready for shipping. Suppose an order arrives to ship 20 units of that item 12 days from now.

Suppose a different customer asks if you can accept an order for 10 units to ship six days from now. Should you accept this order? In other words, do you have 10 units available to promise (ATP) six days from now?

Addressing this question requires the following scheduling parameters:

- Lead time – The time required to obtain the item from your suppliers or manufacturers if you were to place a new purchase order today.
- Inbound processing time – The time required to process a supply after it has arrived in your node.
- Outbound processing time – The time required for your node to assemble an item, perform any value-added services, and physically ship the item.

The sum of these three parameters for an item is known as lead + processing time. Within the lead + processing time, you can expect to

submit a purchase order to your supplier, receive the supply into your node, process the item, and ship it out. Therefore, demands with expected fulfillment dates beyond the lead + processing time can be fulfilled irrespective of your current inventory situation.

To return to the example, you can accept the order for 10 units if the lead + processing time allows you to fill both the order for 10 units six days from now and the order for 20 units 12 days from now.

6.4 Fast-Moving Inventory

In some cases, it may be best not to guarantee inventory availability beyond a certain date even though you currently have sufficient inventory available. If the item in question is a fast-moving item and you can expect to exhaust all on-hand inventory long before an order is to be shipped, you may not want to hold inventory for that future order.

A good example of this can be found in the recent launching of a popular toy. Retailers refused to guarantee availability two weeks from the launch date. They correctly assumed that they would exhaust all available and expected inventory long before orders were due to be shipped.

6.5 Life Span of Supply and Demand

A supply of items is good only while the items are consumable. Some items degrade over time. Examples include pharmaceuticals, food stuffs, beverages, and flowers. The time period during which an item can be kept stocked for consumption is known as its shelf life. The expiration date marks the end of the shelf life of an item.

The time window in which current supply can be used to fulfill demand is referred to as the life span. The date beyond which a supply can no longer be shipped to customers because supply has degraded is called the ship-by date. The ship-by date is always earlier than or equal to the expiration date of the supply.

Selling and Fulfillment Foundation allows users to specify a minimum ship-by date during order creation, which takes into consideration the preferred remaining life span of a time-sensitive item at the time of its shipment during the promising and scheduling processes. For more information about minimum ship-by dates, see Section 9.2.1.3, "Minimum Ship-By Date".

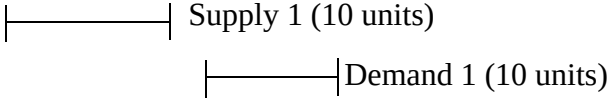
Demand has a life span too. For example, a customer may order a product today contingent on delivery within two weeks from the date of the order. The life span of the demand is two weeks.

One of the most important aspects of inventory management is managing and understanding the life span of supplies and demands.

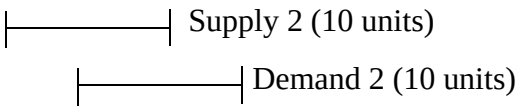
A demand is fulfilled if and only if the following are true:

- The quantity of the supplies is sufficient to fulfill demand.
- The life span of one or more supplies overlaps the life span of the demand.

Consider the following example:



The life span of Supply 1 does not overlap with the life span of Demand 1. Therefore, the demand remains unfulfilled. Demand 1 cannot be fulfilled because there is no available supply to cater to it and a freshly ordered supply does not arrive within the lead + processing time.



The life span of Supply 2 overlaps with the life span of its demand. Therefore, Demand 2 can be fulfilled.

6.6 Inventory Consolidation

The Hub organization specifies the rule to determine how inventory items are identified and consolidated. The Hub can choose to consolidate at either the Hub level or the Enterprise level. When inventory is consolidated at the Hub level, the item ID and unit of measure is assumed to be unique across all organizations and there is no inventory segmentation. Choosing Enterprise level consolidation implies inventory is segregated by each enterprise. For more information about choosing Hub-level or Enterprise-level consolidation, see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide*.

When deciding what level to consolidate inventory at, you must take into consideration the following system functions and how they are affected:

- Drop shipping - You must consider which organizations can be used for drop shipping the sales orders of an organization. Any organization can drop ship their sales orders from any other organization that is part of the same inventory consolidation.

  Note: You can drop ship orders to a ship node whose owner belongs to a different inventory consolidator if you first ensure all inventory adjustments for this ship node are created passing the correct organization code to the adjustInventory API.

  For example, if ShipNode1 is owned by organization code DEFAULT (Inventory Consolidator = DEFAULT), by default if you call adjust Inventory for this ship node (without passing the organization code), the adjustment is made for the inventory consolidator of the ship node owner, in this case DEFAULT.

  If Enterprise1 (Inventory Consolidator = Enterprise1) wishes to source inventory from ShipNode1, all inventory adjustments at ShipNode1 for Enterprise1 must be made by passing OrganizationCode=Enterprise1 when calling the adjustInventory API.

- Inventory visibility - All organizations that are part of the same inventory consolidation have access to inventory of all other

organizations with the only limitation being their own distribution setup.

- Inventory monitoring - Inventory and availability monitoring can only be run for the inventory consolidator.

6.6.1 Hub Level Consolidation

Hub level consolidation provides the most flexibility in drop shipping orders. Any organization can use the shipping node of any other organization to drop ship their sales order. Based on the distribution rules setup of the organization, they can look for item availability across all organizations within the hub.

Choosing Hub level consolidation also exposes the inventory of an organization to all of the other organizations in the Hub and allows the inventory monitor, availability monitor, and FEFO rules to be set only at the Hub level.

6.6.2 Enterprise Level Consolidation

Enterprise level consolidation provides restricted drop shipping for orders. Any organization can use the node of any other organization to drop ship their sales order as long as they all have the same inventory consolidator. Based on the distribution rules setup of the organization, they can look for item availability across all organizations with the same primary Enterprise.

Since inventory is separated by each Enterprise, an Enterprise has no visibility to the inventory of the other Enterprises. Inventory can also be monitored for each Enterprise.

Also, though an organization can participate in multiple Enterprises, they must choose one Enterprise as their inventory consolidator and is the only Enterprise they can maintain inventory in.

6.7 Synchronizing with Node Inventory

Inventory changes can occur in warehouses without Selling and Fulfillment Foundation being immediately aware of it. For example, in the event of lost or damaged goods. Because of this, Selling and Fulfillment Foundation needs to regularly reconcile its internal inventory picture with

the inventory picture at the nodes. This process is broken down into two steps:

1. Loading the inventory picture from a node

   The inventory picture at the node is downloaded to Selling and Fulfillment Foundation. That information is stored in temporary table.

2. Synchronizing the Selling and Fulfillment Foundation inventory picture and the actual inventory picture.

   The data in between the temporary table and the base inventory table is compared, and if needed, synchronized.

Once this process is complete, an agent needs to be configured and run in order to clean up the temporary tables. For more information about configuring the synchronization with node inventory, see the *Sterling Global Inventory Visibility: Configuration Guide*.

6.8 Synchronizing with Node Demand

Selling and Fulfillment Foundation provides the ability to synchronize demands from external systems. Selling and Fulfillment Foundation takes a set of ship nodes, demand types, and demand pictures as input. It then resets the demand information for the ship nodes and demand types based upon the demand picture it received as input.

6.9 Inventory Costing

Selling and Fulfillment Foundation provides the capability to maintain inventory costs against specific products assuming that the costs are passed through Selling and Fulfillment Foundation from financial or procurement systems. Selling and Fulfillment Foundation can also provide a method for determining the cost of returned items. By maintaining inventory costs, Selling and Fulfillment Foundation can pass cost information to financial systems and act as a repository for "cost-centric" reporting.

Selling and Fulfillment Foundation can be set up to maintain inventory costs on a node-by-node basis. For each node, Selling and Fulfillment Foundation maintains records for all inventory receipts at the purchase order, item, and node level. Selling and Fulfillment Foundation also maintains records for shipments out of the node.

Selling and Fulfillment Foundation provides two inventory costing methods:

- Average Costing
- First In First Out (FIFO)

Inventory Value is the total monetary value of all on-hand inventory at a given location, based on per unit Average Cost of each item. It is updated whenever inventory is moved in to or out of an on-hand classification.

Selling and Fulfillment Foundation calculates the new inventory value using the following formula:

```
Old Inventory Value + Change in Inventory Value Due to
Transaction
```

where “Change in Inventory Value Due to Transaction”:

1. Can be positive or negative (depending on whether the transaction represents a receipt or issue from on-hand inventory)
2. Is calculated from transaction quantity * unit value of the transaction document

Selling and Fulfillment Foundation provides integration into financial applications to vary ledger account determination by something such as product line or department. The `Posting Classification` attribute specified during Item definition is used to group items into appropriate Financial ledger groups. When translating a transaction from Selling and Fulfillment Foundation into a series of financial postings, the classification value for the item in the transaction is carried in the interface.

6.9.1 Average Costing

For the Average Costing method, inventory adjustments accumulate for manual review by a cost accountant before posting records to the matching tables. This allows cost accountants to adjust cost at the aggregate item level and cancel out adjustments.

The Selling and Fulfillment Foundation Average Costing functionality includes:

- Defining inventory costs within Selling and Fulfillment Foundation

- Capturing relevant costs during purchase order execution and sales execution
- Ensuring that the correct and most current costs are passed from Selling and Fulfillment Foundation to a target financial (A/P, A/R and G/L) application
- The ability to view the current monetary value of inventory within Selling and Fulfillment Foundation at any point in time

6.9.1.1 Determining Unit Costs

Unit cost represents the normal or specified cost used as the basis for measurement against an actual. Unit costs for manufactured items include labor, material and overhead, vendor acquisition, freight, duty fees and other categories for purchased items. Unit cost for physical kits is determined by the sum of the unit costs for all components of the kit PLUS the WIP Cost Factor defined for the item in the item master. If the WIP Cost Factor is not available on the item master, the WIP Cost Factor defined for the Primary Enterprise of the catalog organization is used.

Unit cost is stored differently depending on the base cost used for its calculation.

Selling and Fulfillment Foundation provide the capability to define unit costs based on the following methods for determining inventory value:

- Replacement
- Average

6.9.1.1.1 Replacement Cost

Replacement Cost is the per-unit cost of acquiring inventory from a given supplier and is used as the default cost on purchase orders. Selling and Fulfillment Foundation represents replacement cost as the published price list of the supplier.

Replacement Cost is stored within Selling and Fulfillment Foundation in a structure that can vary the per unit cost by:

- Supplier
- Date

This is called the Vendor Price List.

The Selling and Fulfillment Foundation provides the capability to use replacement cost in the following ways:

- As the default unit cost on purchase order lines. This can be overridden manually.
- As the base cost for unit cost calculations.

6.9.1.1.2 Replacement Cost Used as the Base Cost

Unit cost is stored at the item level. In the current release, there is no means provided to keep replacement cost at supplier-location level and hence replacement cost does not vary by each supplier shipping location. Since replacement cost is used as the basis, unit cost does not vary by each location and can be kept at the item level.

Selling and Fulfillment Foundation provides a mechanism to re-compute the unit cost for an item through an API. This API can be invoked whenever there is a change made to replacement cost (or any other time needed) and the Selling and Fulfillment Foundation cost determination logic re-determines the unit cost based on the factors applicable. Selling and Fulfillment Foundation does not automatically re-compute the unit cost when replacement cost changes and currently an external trigger needs to invoke the API for re-computing the unit cost.

When having to resolve the source of the base replacement cost, it is necessary to identify the primary source of this item in the `Primary Supplier` attribute during item definition. Selling and Fulfillment Foundation supports items that are sourced from more than one location. This nomination indicates the default source from which Selling and Fulfillment Foundation can determine the list of cost factors when procured from a supplier.

When using replacement cost as the base cost, the unit cost is re-computed as:

Unit cost + Landed Cost Factor + Standard Cost Factor (applied after Landed Cost Factor added) = Computed Unit Cost

Where:

- Unit cost is the unit cost from the price list of the primary supplier.
- Standard Cost Factor is the Standard Cost Factor of the primary supplier.

- Landed Cost Factor is the Landed Cost Factor of the primary supplier. If either the Standard or Landed Cost Factor of the primary supplier is not defined, the Cost Factor defined for the Primary Enterprise of the primary supplier is used.

For more information about cost factors, see Section 6.9.1.2, "Cost Factors".

6.9.1.1.3 Average Cost

The average cost is the cost of an item at a specific location PLUS in-bound costs such as freight. Average cost is revalued when:

- Product is received, against a purchase order
- Or finished goods are received through a "production" operation.

6.9.1.1.4 Average Cost Used as the Base Cost

Unit cost is recorded at the item and location (the Selling and Fulfillment Foundation ship node) level. It is always expressed as an amount at the item and location level.

Re-computation of unit cost is done automatically by the system whenever there is a change made to average cost. No external trigger is required for this case. Note the difference between this and when replacement cost is used as the base cost.

6.9.1.1.5 Average Cost Calculation – PO Receipt

Average Cost is calculated upon receipt of a PO using the following:

(Total On Hand Inventory Value PLUS Total Landed Value Received) divided by (New Total Quantity On Hand)

Where:

- Inventory Value has been recalculated
- Total Landed Value equals Expected unit price * Quantity received + Additional cost based on associated cost factors.
- The unit price on the purchase order becomes the cost for the buyer.

Note: Even though the Selling and Fulfillment Foundation Supply Collaboration application allows you to specify charges other than unit cost, only the unit price specified in the purchase order is used for calculation of the total value received. Additional costs are calculated based on cost loading factors applicable for the receipt or supplier.

The following example shows the calculation of average cost during a purchase order receipt. The Cost Factor Group referenced in this example is defined in Example 6–2. In this example, the Cost Factor Group is associated with the vendor. Each unit of the item weighs 1 pound.

Example 6–1 Average Cost Calculation - PO Receipt

Current on-hand quantity before receipt = 500.

Inventory value before receipt = $2000.

A purchase order line for 1000 units of an item has the following prices defined:

Expected unit cost = $3.00.

Total purchase order line cost = $3000.

Additional cost based on the associated Cost Factor Group is calculated as:

Freight	**(2.5%)**	**0.025 * 3000 = $75.00**
Duty	(0.05 / pound)	1000 * 1 * .05 = $50.00
Insurance	(0.75%)	0.0075 * 3000 = $22.50
Brokerage	(1%)	0.01 * 3000 = $30.00

Total additional cost: $177.50

When a receipt for 1000 units for this purchase order line is made, the average cost is computed as:

($2000 + $3000 + $177.50) / (500 + 1000) = $3.451667

When the receipt of the PO line item is published for general ledger posting, Selling and Fulfillment Foundation provides the breakup of inventory value as:

Extended cost: $3000

Freight: $75.00

Duty: $50.00

Insurance: $22.50

Brokerage: $30.00

Total: $3177.50

The breakup provided can be used to post to the appropriate general ledger accounts.

6.9.1.1.6 Average Cost Calculation – Work Order Completion

Average Cost is calculated upon receipt of a work order completion using the following:

(On-hand Inventory value + Total value of finished item) divided by (New on hand quantity)

Where:

- Inventory Value has been recalculated
- Total value of the Finished Item is equal to:
  - Sum of average cost of all components + Additional cost as calculated based on associated cost factor group.
  - A cost factor group can be defined at enterprise level or parent item level for work orders. For a given business unit (enterprise), only one such group can be defined.

The following example shows the calculation of average cost during a work order completion. The Cost Factor Group referenced in this example is defined in Example 6–5. In this example, the Cost Factor Group is associated with the Enterprise.

Current on-hand quantity before work order completion = 500.

Inventory value before completion = $5000.

A work order for 1000 units of the finished good is received.

The finished good is made up of the following components:

- 12345 – Average cost = $2.10
- 32456 – Average cost = $6.20

When a work order for 1000 units for this finished good is completed, the total value of the finished item is calculated as:

Component cost		(1000 * $2.10) + (1000*6.20) = $8300
Labor Cost	(20%)	0.2 * 8300 = $1660
Supplies	($3.00/unit)	$3000

Total increase in inventory value = $12,960.00

The new Average cost is calculated as:

($5000 + $12960) / (500 + 1000) = $11.973333

6.9.1.1.7 Handling Negative Inventory Balances

When doing a receipt against an item or node that has a negative on-hand balance, Inventory Value and Average Cost calculations are modified as follows:

1. Average cost is set to the loaded cost of the current transaction. The loaded cost is the unit price on the PO PLUS the Landed Cost Factors for the seller on the PO.
2. Inventory Value is set to the value of on-hand inventory * new average cost.
3. Selling and Fulfillment Foundation generates a second event to accompany the standard inventory value change. This second event publishes the delta between the recalculated inventory value and the write-off amount which is calculated as:

   Old Inventory Value + Change in Inventory Value – New Final Inventory Value

For example:

A product currently has an average cost of $3.50. The on-hand quantity in the system is -20 and the inventory value is -$70. When a receipt is created for 15 units of this item with a loaded cost of new

receipt as $4.00, Selling and Fulfillment Foundation updates inventory as:

Average cost = $4.00 – loaded cost from the receipt

On hand quantity = -5

Inventory value = -$20.00

The standard inventory event publishes an increase of $60.00 to the inventory value of the item. This increase includes the loaded cost of $4.00 * 15 (the number of units received).

An additional inventory write-off event publishes the write-off amount as $10.00 calculated as (Old Inventory value + Change in inventory value due to the receipt – New final inventory value) or (-$70 + $60 – (-$20) = $10.00. In this instance, an entry to the financial application is created with a debit to an adjustment account and a credit to an inventory account. If the result were a negative amount, the entry would be credit to an adjustment account and debit to an inventory account. Selling and Fulfillment Foundation ensures that this is represented as such to the financial application.

6.9.1.2 Cost Factors

Cost factors represent a value modifier that is an additional function or component from a base cost to give a new unit cost. Examples of cost factors include insurance, freight, material handling, and packaging. These activities represent added value relative to the base point that an organization must track to give its derived cost. For example, acquisition from a vendor.

Selling and Fulfillment Foundation enables cost factors and the definition of unit cost as a relationship between a nominated value, such as replacement cost, and one or more cost factors.

Selling and Fulfillment Foundation uses cost factors at the following points to arrive at the derived cost:

- Calculation of loaded cost of inventory during the receipt process.
- Calculation of loaded cost of kitted finished good items when the work order is completed
- Calculation of unit cost from replacement cost or average cost

6.9.1.2.1 Cost Factor Definition

An organization can define many cost factors. Each cost factor has the following attributes:

Name	Unique Name for Cost Factor
Calculation method - Percent or Value?	Is this factor expressed as a percentage or an amount?
Application method	If the calculation method is defined as "Percentage", the application method has no significance. If the calculation method is specified as “Amount”, the application method can have one of the following values: • Weight – the value specified represents the dollar value for each pound of product. The base weight can be defined as any of weight UOM. • Volume - The value specified represents the dollar value for each cubic foot of product. The base volume can be defined as any of weight UOM. • Quantity – The value specified represents the monetary value for each unit of the product.

6.9.1.2.2 Cost Factor Groups

Since there may be situations when different cost factors need to be applied based on vendor or transaction type (such as work order completion or receipt of product), Selling and Fulfillment Foundation allows the creation of a *Cost Factor Group*. A cost factor group represents a set of *cost factors* that are applied to the base cost for a given scenario. Each cost factor within a group is a reference to a common definition – the group gives a specific value to a factor for the particular scenario.

6.9.1.2.3 Assignment of Cost Factor Groups

To provide maximum flexibility and minimize maintenance, a hierarchy approach is provided for assigning and retrieving cost factors.

Level 0: Enterprise Level

Level 1: Supplier Level

Level 2: Item Level – For physical kit parent items in WIP processing ONLY

At each level an organization can assign a series of Cost Factors to different transactions and assign a value to each cost factor.

6.9.1.2.4 Cost Factor Retrieval

During cost factor retrieval, the application begins at level 2 and works back to level 0 of the assignment hierarchy to retrieve the list of cost factors most specific to the particular transaction. Cost factors are retrieved from a single level only.

6.9.1.2.5 Cost Factor Group Examples

Example 6–2 through Example 6–6 illustrate how you might set up cost factor groups for specific situations.

Example 6–2 Cost Factor Group For Product Imported From International Vendors

For this group, the following cost loading factors are specified:

Cost factor name	Calculation method	Application Method	Percentage	Value	Comment
Freight	Percentage	-	2.5%	-	Freight is calculated as percentage of base cost
Duty	Amount	Weight	-	0.050000	Duty is calculated as 0.05 / pound of product
Brokerage	Percentage	-	1%	-	Brokerage is 1% of the product cost
Insurance	Percentage	-	0.75%	-	Insurance is calculated as% of base cost

Example 6–3 Cost Factor Group For Product Sourced From Domestic Vendor

For this group, the following cost loading factors are specified:

Cost factor name	Calculation method	Application Method	Percentage	Value	Comment
Freight	Percentage	-	2%	-	Freight is calculated as percentage of base cost
Insurance	Percentage	-	0.0050	-	Insurance is calculated as% of base cost

Example 6–4 Cost Factor Group For Product Sourced From Domestic Vendor Who Include Freight In Their Unit Price

For this group, the following cost loading factors are specified:

Cost factor name	Calculation method	Application Method	Percentage	Value	Comment
Insurance	Percentage	-	0.0050	-	Insurance is calculated as% of base cost

Example 6–5 Kitted Finished Product Cost

For this group, the following cost factors are specified:

Cost factor name	Calculation method	Application Method	Percentage	Value	Comment
Labor	Percentage	-	20%		20% of the base cost is added to arrive at the labor cost
Supplies	Amount	Quantity	-	3.00	$3.00 is added to every finished kitted unit as the supplies cost

Example 6–6 Unit Cost Calculation From Replacement Cost

For this group, the following cost loading factors are specified:

Cost loading factor name	Calculation method	Application Method	Percentage	Value	Comment
Handling	Percentage	-	5%		5% of the base cost is added to arrive at the derived cost

6.9.1.3 Inventory Costing for Stocked Products

The following events effect cost elements or general ledger postings when handling stocked products and each event is discussed in Section 6.9.1.3.1 through Section 6.9.1.3.4.

1. Events related to purchase orders (see Section 6.9.1.3.1)
   a. Purchase order creation
   b. Purchase order update
   c. Receipt of purchase order
   d. Invoice matching function performed in the financial system
2. Events related to sales order (see Section 6.9.1.3.2)
   a. Sales order creation
   b. Shipment confirmation
   c. Publishing of invoice
3. Adjustment of inventory (see Section 6.9.1.3.3)
4. Returns of goods (see Section 6.9.1.3.3)
5. Completion of work order (see Section 6.9.1.3.4)

6.9.1.3.1 Processing a Standard Purchase Order

Selling and Fulfillment Foundation processes a standard Purchase Order as follows:

PO Creation and Update

1. Selling and Fulfillment Foundation includes the price on a PO line from the Vendor Price Lists maintained in Selling and Fulfillment Foundation.

   If vendor prices are not maintained in Selling and Fulfillment Foundation, the price may be retrieved from an external vendor pricing application.

2. Selling and Fulfillment Foundation provides the ability to lock a vendor unit price on a PO line.

   If the price list of a vendor was used for unit price calculations, Selling and Fulfillment Foundation recalculates the unit price whenever there is any quantity change made to the purchase order line. This is done so that the appropriate quantity break-up can be used for unit price. If the PO unit price was overridden on the purchase order, Selling and Fulfillment Foundation skips the calculation and recalculates only the extended cost of the line.

3. Selling and Fulfillment Foundation raises a standard event on the successful creation and update of a purchase order. This can be used to update the financial application with expected purchase order cost related information.

Receipt of PO

4. Selling and Fulfillment Foundation increases inventory value upon receipt of product against a PO. This calculation is used to determine the Inventory Value.

5. Selling and Fulfillment Foundation revalues the average cost of an item at a node when product is received against a PO. This calculation is used to determine the Average Cost.

6. Selling and Fulfillment Foundation publishes the data for each receipt line. This is used to generate general ledger level postings in a financial application. For more information about the inventory transactions and the specific data published, see the *Selling and*

Fulfillment Foundation: Application Platform Configuration Guide for time triggered transaction appendix.

7. As each PO line can have more than one receipt recorded against it in Selling and Fulfillment Foundation, it is possible for a PO line to be included in multiple general ledger interface records – one for each receipt line. Therefore, one event is published for each purchase order line as a receipt is recorded against it. If a line is received in multiple receipts, multiple events are raised.

8. When a receipt is made against an existing on-hand balance that is less than zero, Selling and Fulfillment Foundation uses the special handling described in Section 6.9.1.1.7.

9. Selling and Fulfillment Foundation differentiates between the processing of standard Purchase Orders and chained Purchase Orders.

   For example, the A/P interfacing of a chained PO MAY need to be suppressed when the notification of fulfillment is signaled by the supplier submitting their invoice.

Invoice Creation

10. The Invoice matching function is performed in the financial application as follows:

    a. Discrepancies found during invoice matching function are posted in variance accounts other than the Inventory value account.

    b. If the Accounts Payable application generates a variance between the expected PO cost and the actual cost on the Payable Invoice, the variance must be passed back to Selling and Fulfillment Foundation to be reflected in average cost. Selling and Fulfillment Foundation provides an API (updateInventoryCost) that accepts the variance amount (plus PO reference and quantity invoiced) and attempts to adjust the inventory value by this amount. If the total on-hand is less than what was invoiced (due to subsequent shipments or issues), the total variance is prorated and applied to the remaining on-hand inventory. The amount not applied is passed back to the financial application so that it can be stored in an appropriate variance account.

Standard Purchase Order Scenarios

Note: Though not illustrated in these scenarios, cost factor retrieval is a standard part of cost calculation.

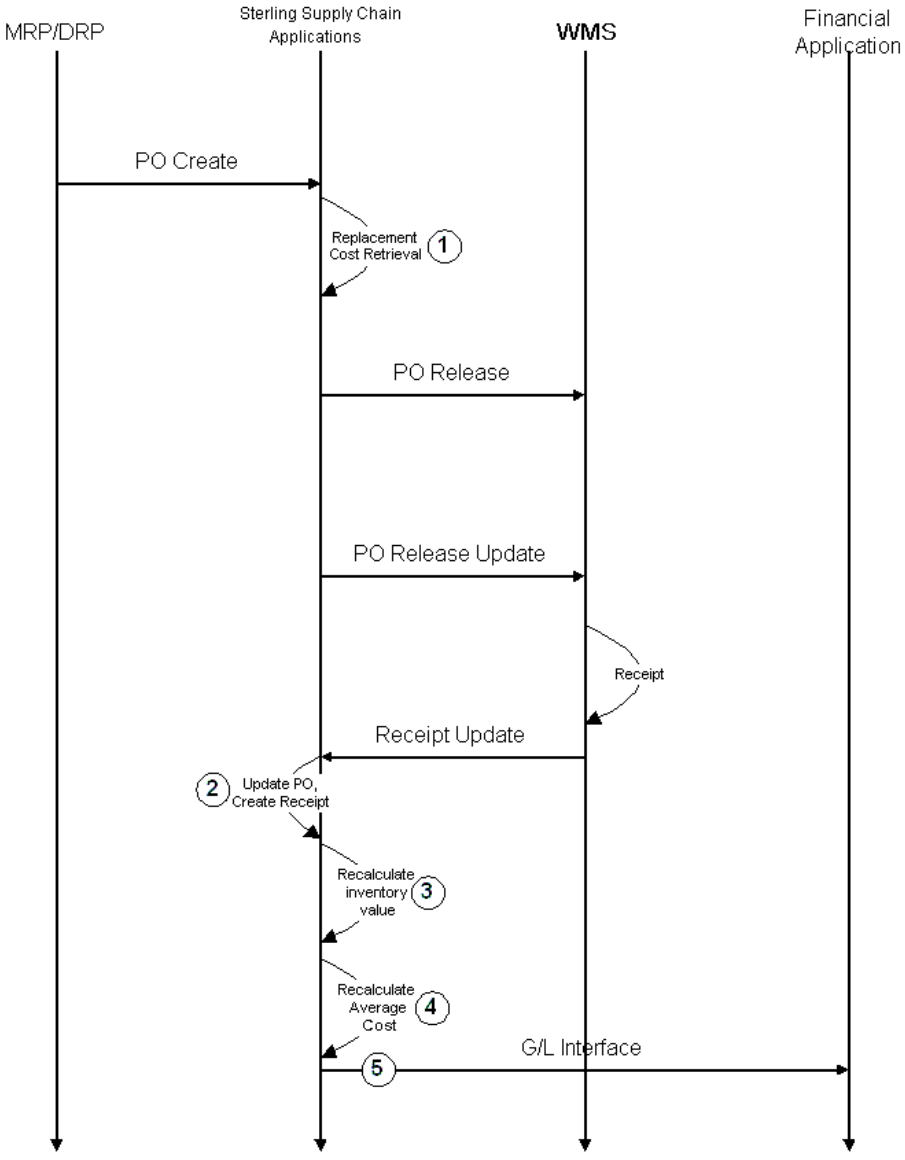
Example 6–7 Standard PO - Scenario 1 – See Figure 6–3

A purchase order is created for item ABC. The PO creation message has identified the expected supplier (S1) of ABC for this particular transaction. A quantity of 50 pieces is required. On entry, the replacement cost for buying ABC from S1 is retrieved and stamped on the PO as the line price. A quantity of 50 pieces of ABC is received against the PO line (or ASN). For each receipt the value of inventory at the receiving location is recalculated and a new average cost, based on the received quantity and expected cost of the PO line, is determined. Finally, the information about the receipt (quantity received, item, expected price) is sent to the financial application.

1. Selling and Fulfillment Foundation perform Replacement Cost Retrieval
   a. Criteria used: Order Date, Supplier, Item, Order Quantity
   b. A user exit allows the standard Selling and Fulfillment Foundation retrieval to be bypassed
   c. If no user exit is implemented, Selling and Fulfillment Foundation retrieves the price from the specified Price List
   d. Selling and Fulfillment Foundation updates the PO line with expected cost – from either the user exit invocation or retrieval from Selling and Fulfillment Foundation
   e. Result PO price = $2.00 each piece
2. Each receipt line transaction in Selling and Fulfillment Foundation performs the steps 3 through 5.
3. Selling and Fulfillment Foundation recalculates the Inventory Value
   a. Current Inventory Value = $250
   b. Value of new receipt = 50*2.00 = $100
   c. New Inventory Value = $350

4. Selling and Fulfillment Foundation recalculates the Average Cost
   a. On-Hand Inventory (pre-receipt) = 120
   b. On-Hand Inventory (post-receipt) = 170
   c. New Inventory Value = $350
   d. New Average Cost = 350/170 = $2.058824
5. Selling and Fulfillment Foundation publishes the change in inventory value (used for integration to financial applications). For more information about the inventory transactions and the specific data published, see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide* for time triggered transaction appendix.

Figure 6–3 Purchase Order Scenario 1



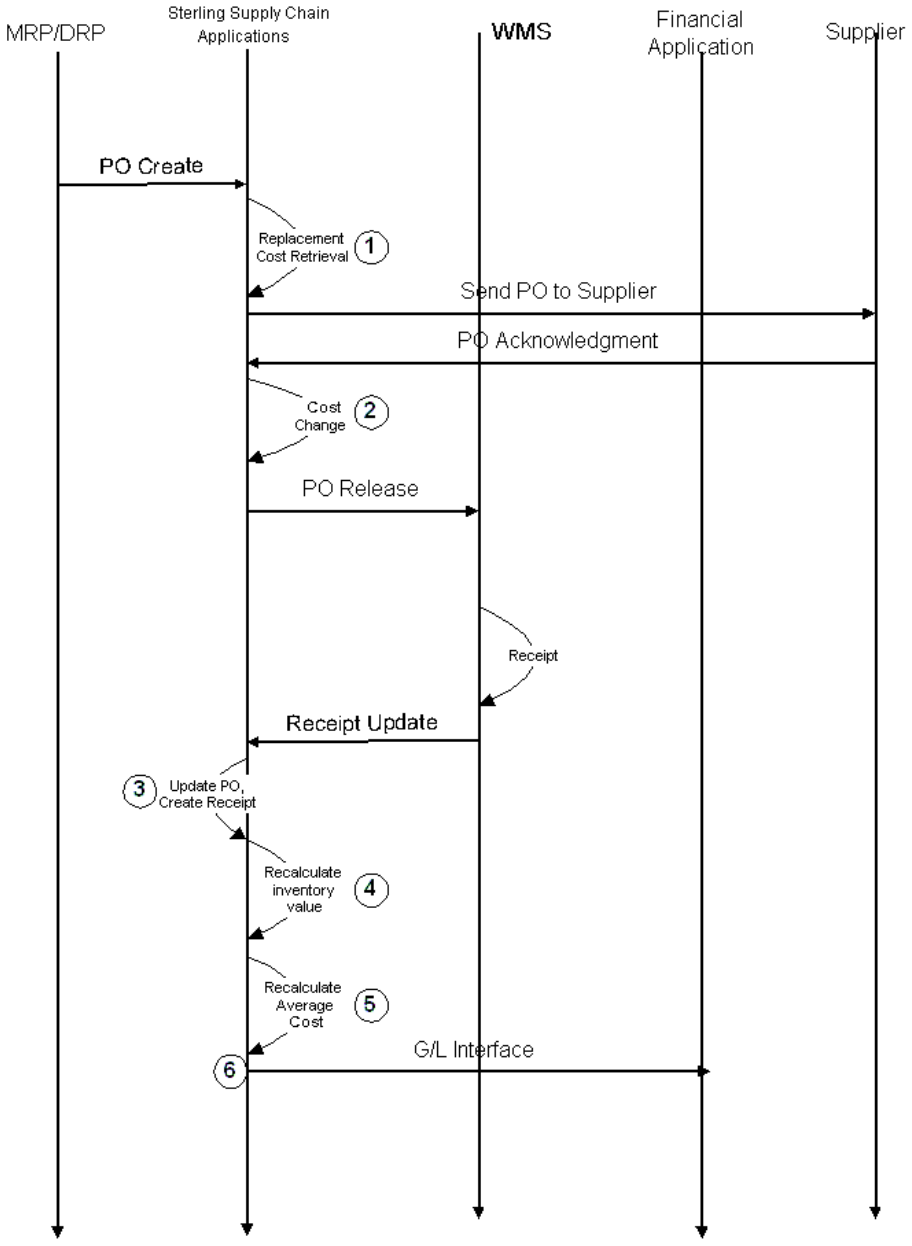
Example 6–8 Standard PO - Scenario 2 – See Figure 6–4

A purchase order is created for item ABC. The PO creation message has identified the expected supplier (S1) of ABC for this particular transaction. A quantity of 50 pieces is required. On entry the replacement cost for buying ABC from S1 is retrieved and stamped on the PO as the line price. On receipt of the PO acknowledgement from S1, it is noted that the per-unit price has been increased. This new price is entered on the original PO as a PO change. A quantity of 50 pieces of ABC is received against the PO line/Shipment. For each receipt the value of inventory at the receiving location is recalculated and a new average cost, based on the received quantity and expected cost of the PO line, is determined. Finally the information about the receipt (quantity received, item, expected price) is sent to the financial application.

1. Selling and Fulfillment Foundation performs Replacement Cost Retrieval
   a. Criteria used: Order Date, Supplier, Item, Order Quantity
   b. A user exit allows the standard Selling and Fulfillment Foundation retrieval to be bypassed
   c. If no user exit is implemented, Selling and Fulfillment Foundation retrieves the price from the specified Price List
   d. Selling and Fulfillment Foundation updates the PO line with expected cost – from either the user exit invocation or retrieval from Selling and Fulfillment Foundation
   e. Result PO price = $2.00 each piece
2. Selling and Fulfillment Foundation updates the PO Price from Supplier
   a. Price from inbound message must change PO price
   b. New price = $2.10
3. Each receipt line transaction in Selling and Fulfillment Foundation performs steps 2 through 6.
4. Selling and Fulfillment Foundation recalculates the Inventory Value
   a. Current Inventory Value = $250
   b. Value of new receipt = 50*2.10 = $105
   c. New Inventory Value = $355

5. Selling and Fulfillment Foundation recalculate the Average Cost
   a. On-Hand Inventory (pre-receipt) = 120
   b. On-Hand Inventory (post-receipt) = 170
   c. New Inventory Value = $355
   d. New Average Cost = 355/170 = $2.088235
6. Selling and Fulfillment Foundation publishes the change in inventory value (used for integration to financial applications). For more information about the inventory transactions and the specific data published, see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide* for time triggered transaction appendix.

Figure 6–4 Purchase Order Scenario 2



Example 6–9 Standard PO - Scenario 3 – See Figure 6–3

A purchase order is created for item ABC. The PO creation message has identified the expected supplier (S1) of ABC for this particular transaction. A quantity of 50 pieces is required. On entry the replacement cost for buying ABC from S1 is retrieved and stamped on the PO as the line price. A quantity of 40 pieces of ABC is received against the PO/ASN. For each receipt the value of inventory at the receiving location is recalculated and a new average cost, based on the received quantity and expected cost of the PO line, is determined. Finally the information about the receipt (quantity received, item, expected price) is sent to the financial application.

1. Selling and Fulfillment Foundation performs Replacement Cost Retrieval
   a. Criteria used: Order Date, Supplier, Item, Order Quantity
   b. A user exit allows the standard Selling and Fulfillment Foundation retrieval to be bypassed
   c. If no user exit is implemented, Selling and Fulfillment Foundation retrieves the price from the specified Price List
   d. Selling and Fulfillment Foundation updates the PO line with expected cost – from either the user exit invocation or retrieval from Selling and Fulfillment Foundation.
   e. Result PO price = $2.00 each piece
2. Each receipt line transaction in Selling and Fulfillment Foundation performs steps 3 through 5.
3. Selling and Fulfillment Foundation recalculates the Inventory Value
   a. Current Inventory Value = $250
   b. Value of new receipt = 40*2.00 = $80
   c. New Inventory Value = $330
4. Selling and Fulfillment Foundation recalculates the Average Cost
   a. On-Hand Inventory (pre-receipt) = 120
   b. On-Hand Inventory (post-receipt) = 160
   c. New Inventory Value = $330
   d. New Average Cost = 330/160 = $2.0625

5. Selling and Fulfillment Foundation publishes the change in inventory value (used for integration to financial applications). For more information about the inventory transactions and the specific data published, see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide* for time triggered transaction appendix.

> **Note:** Any additional receipt transactions that receive the outstanding quantity, follow the above process. But any adjustment to an existing receipt that is done in a Warehouse Management System must be manually entered in the financial application.

Example 6–10 Standard PO - Scenario 5

Item JJJ currently has an average cost of $3.50. The current on-hand quantity in the system is -20 and the inventory value is -$70. A receipt is made for 15 units of JJJ against a PO that has a line price of $4.00.

1. Selling and Fulfillment Foundation sets the average cost to the PO line price (Average cost from $3.50 to $4.00)
2. New on-hand quantity = -5 (-20+15)
3. Selling and Fulfillment Foundation recalculates the new inventory value
   a. Since the on-hand quantity started as negative, Selling and Fulfillment Foundation uses the MODIFIED calculation
   b. New Inventory Value = On-Hand * New Average Cost = -$20
4. Selling and Fulfillment Foundation determines the write-off in inventory value
   a. A standard inventory event publishes an increase of $60.00 (15 * $4) to the inventory value of the item
   b. An additional inventory write-off event publishes the write-off amount as $10.00 calculated as (Old Inventory value + Change in inventory value due to the receipt – New final inventory value) or (-$70 + $60 – (-$20) = $10.00

Example 6–11 Standard PO - Scenario 6

Item JJJ currently has an average cost of $3.50. The current on-hand quantity in the system is -20 and inventory value is -$70. A receipt is made for 25 units of JJJ against a PO that has a line price of $4.00.

1. Selling and Fulfillment Foundation sets the average cost to the PO line price (Average cost from $3.50 to $4.00)
2. New on-hand quantity = 5 (-20+25)
3. Selling and Fulfillment Foundation recalculates the new inventory value
   a. Since the on-hand quantity started as negative, Selling and Fulfillment Foundation uses the MODIFIED calculation
   b. New Inventory Value = On-Hand * New Average Cost = $20
4. Selling and Fulfillment Foundation determines the write-off in inventory value
   a. A standard inventory event publishes an increase of $100.00 (25 *$4) to the inventory value of the item.
   b. An additional inventory write-off event publishes the write-off amount as $10.00 calculated as (Old Inventory value + Change in inventory value due to the receipt – New final inventory value) or (-$70 + $100 – ($20) = $10.00

6.9.1.3.2 Processing a Standard Sales Order

Selling and Fulfillment Foundation process a standard Sales Order as follows:

ORDER CREATION

On a sales order line creation and draft sales order confirmation, Selling and Fulfillment Foundation uses the following logic to retrieve the unit cost of each item:

1. If the unit cost was overridden on the order line, Selling and Fulfillment Foundation uses the override cost as the unit cost.

2. When Replacement Cost is used as the basis for unit cost computation:
   - If the unit cost was manually entered at the item level in the product master tables, the order line uses the manually entered unit cost
   - If no manual entry was made, the order line uses the computed unit cost stored at the item level. If no such cost was stored, the cost is reflected as $0.00 on the sales order line and the ORDER_CREATE.ON_ZERO_UNIT_COST event is triggered.
   - Logical kit unit cost is stored at the component level and is maintained at this level.
3. When Average Cost is used as the basis for unit cost computation: **Note:** this cost basis is not supported in this release.
   - If the unit cost was overridden on the order line, the order line uses the specified cost as the unit cost.
   - A ship node must be identified on the order line for computing the unit cost. If no ship node is specified, unit cost is not stored and shows as $0.00.
   - Whenever a ship node is specified on the order line, Selling and Fulfillment Foundation stores the unit cost from the item-node level. If a change is made to the ship-node of an order line, the unit cost of the order line is picked up from the new node.

SHIPMENT CONFIRMATION

1. When product is shipped, Selling and Fulfillment Foundation recalculates the Inventory Value for the fulfillment location. See the calculation for Inventory Value in Section 6.9.
2. When a shipment is confirmed in Selling and Fulfillment Foundation, Selling and Fulfillment Foundation publishes the information to be used in a Financial application. For more information about the inventory transactions and the specific data published, see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide* for time triggered transaction appendix
3. The Shipment Confirmation event is used to update general ledger entries for cost of goods sold, inventory and various variance accounts. This event is published for each order line separately and

Sales and A/R postings should typically be made through the invoice publishing event.

INVOICE CREATION

4. Selling and Fulfillment Foundation invoice publication interface posts the sales and account receivable general ledger entries.

Standard Sales Order Scenarios

> **Note:** Though not illustrated in these scenarios, cost factor retrieval is a standard part of cost calculation.

Example 6–12 Standard Sales Order - Scenario 1 - See Figure 6–5

A customer places an order for a quantity of 2 pieces of item ABC. The order is priced according to pricing rules for the customer. The expected Unit Cost of ABC is also stamped on the order line as ship node determination has been done during order creation. The fulfillment location reports shipment of ABC, at which time the inventory value change event is published detailing the various cost buckets. When the invoicing transaction runs, the appropriate Financial application interface entries are generated with shipped quantity and sales price.

1. Order Creation
   a. Order is priced according to the price list defined by the seller
   b. Unit Cost is stamped on the order line based on the primary supplier of ABC.
2. Shipment Confirmation

   The average cost of item ABC and the fulfillment location is recorded for the shipment or order line.
3. Selling and Fulfillment Foundation recalculates the Inventory Value
   a. Current Inventory Value = $250
   b. Current Average Cost = $2.05725
   c. Value of shipment = 2*2.05725 = $4.1145
   d. New Inventory Value = $245.8855

e. Selling and Fulfillment Foundation publishes the Inventory Change information (used for integration to financial applications). For more information about the inventory transactions and the specific data published, see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide* for time triggered transaction appendix.

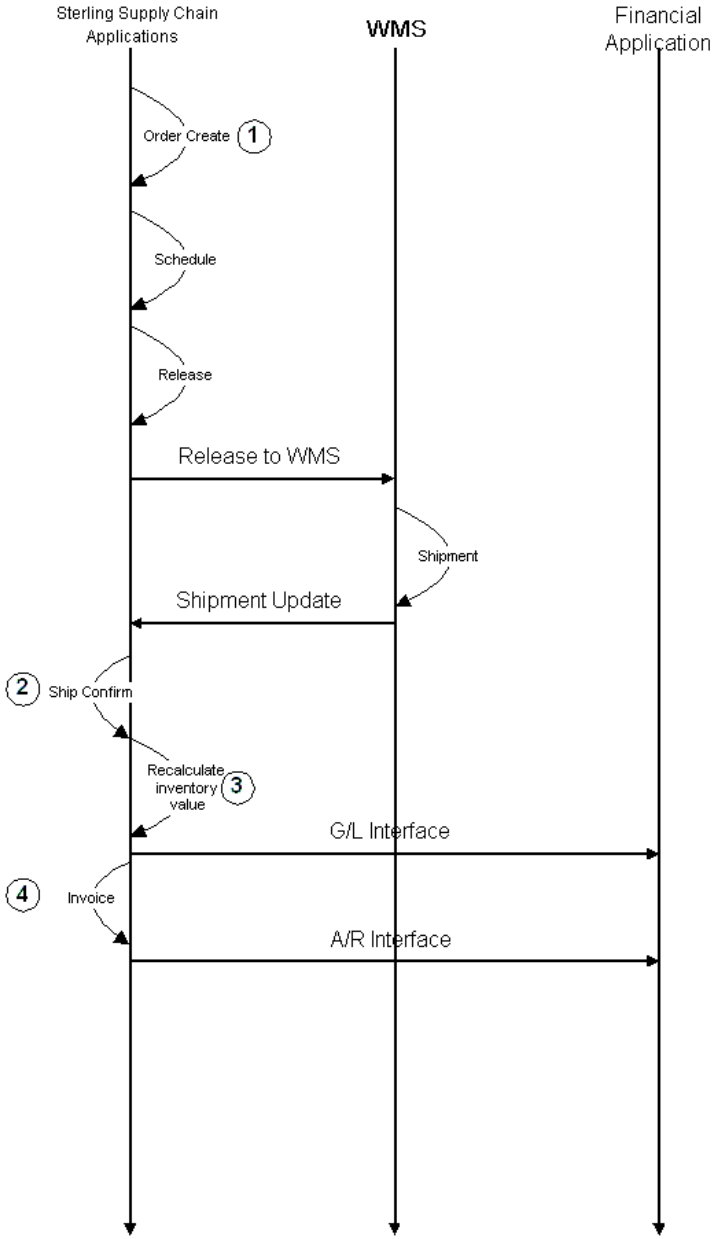
4. Invoice Creation

   a. The sales order invoice does not publish any cost related data

   b. All cost-related data recorded during shipment confirmation (step 2) is passed

   c. An invoice can trace to the contained shipments through shipment number to obtain the cost data if needed. The invoice carries all price related information about the transaction that is used to form the corresponding Accounts Receivable entries.

Figure 6–5 Sales Order Scenario 1



Example 6–13 Standard Sales Order - Scenario 2 - See Figure 6–6

A customer places an order for a quantity of 10 pieces of item ABC. The order is priced according to pricing rules for the customer. The expected Unit Cost of ABC is also stamped on the order line. The order is scheduled, and it is determined that fulfillment is from Node1 (quantity=7) and Node2 (quantity=3). The order is released. The fulfillment locations report shipment of ABC, at which time the inventory value change event is published detailing the various cost buckets. When the invoicing transaction runs (using SHIPMENT based invoicing), the appropriate Financial application interface entries are generated with shipped quantity and sales price.

1. Order Creation
   a. The Order is priced according to the price list defined by the seller
   b. Unit Cost is stamped on the order line based on the primary supplier of ABC
2. Schedule

   The Order Line is split into two schedule entries to represent 2 fulfillment sources – Node1 and Node2
3. Release

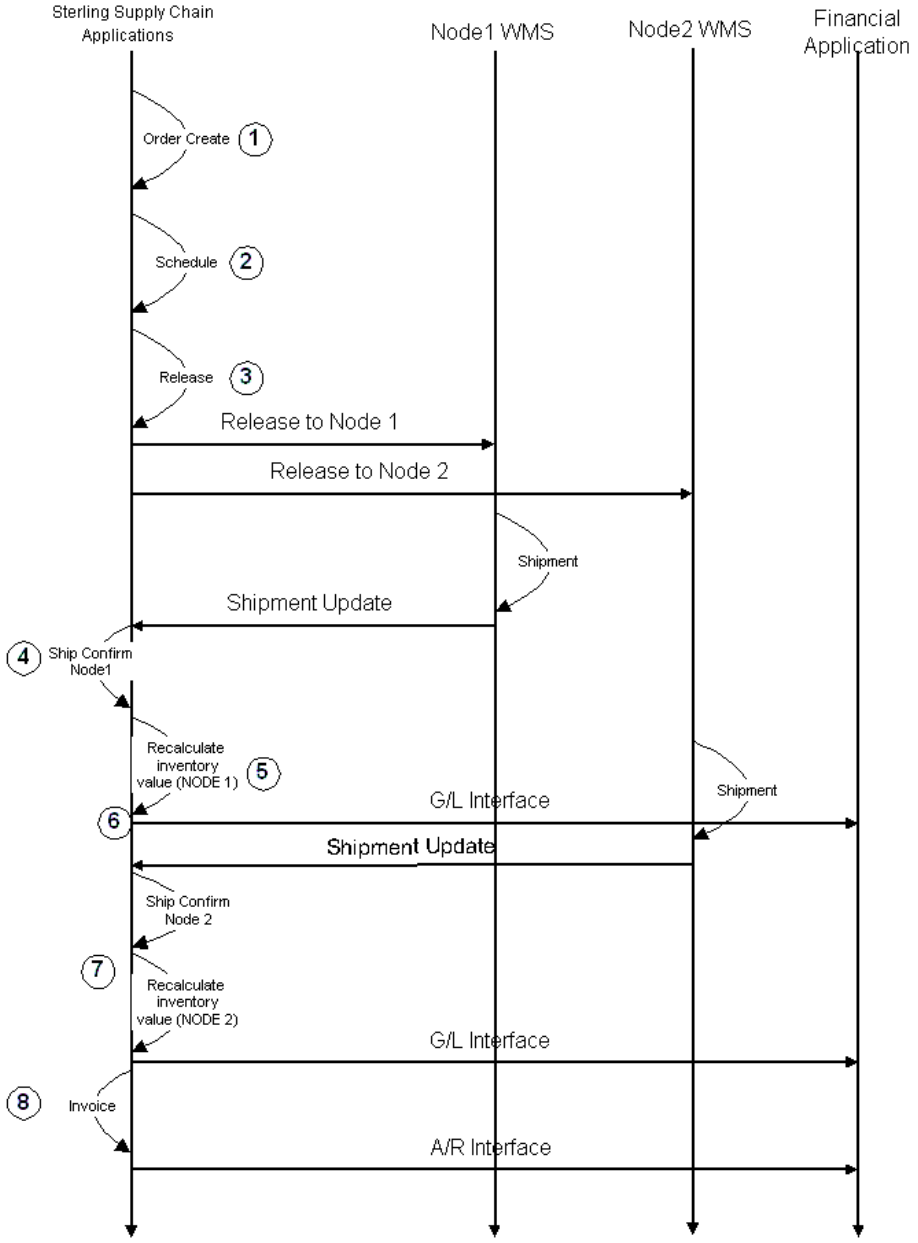
   Selling and Fulfillment Foundation creates an order release for each node to carry out against the order line
4. Shipment Confirmation – Node1

   Selling and Fulfillment Foundation records the average cost of item ABC/fulfillment location for the shipment or order line.
5. Selling and Fulfillment Foundation recalculates the Inventory Value for Node1
6. Selling and Fulfillment Foundation publishes the Inventory Change information (used for integration to financial applications). For more information about the inventory transactions and the specific data published, see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide* for time triggered transaction appendix.
7. Selling and Fulfillment Foundation repeats steps 4-6 for Node2
8. Invoice Creation
   a. The sales order invoice does not publish any cost related data

b. All cost-related data recorded during shipment confirmation (step 4) is passed.

c. An invoice can trace to the contained shipments through shipment number to obtain the cost data if needed. The invoice carries all price related information about the transaction that is used to form the corresponding Accounts Receivable entries.

Figure 6–6 Sales Order Scenario 2



6.9.1.3.3 Processing Returns and Inventory Adjustments

Selling and Fulfillment Foundation processes Return Orders and makes inventory adjustments as follows:

GENERAL

1. Returns processing does not assume the use of the Selling and Fulfillment Foundation Reverse Logistics module. The inventory adjustment can just be flagged as a "Return" type.
2. The processing of returns and inventory adjustments do not force a recalculation of average cost.

   Note: The updateInventoryCost API can be used to recalculate the inventory value. For information about the updateInventoryCost API, see the *Selling and Fulfillment Foundation: Javadocs*.

3. Both returns and inventory adjustments force a recalculation of inventory value, based on the quantity received or issued.
   - The return receipt impacts an on-hand supply type. Supply types are configurable by the organization. This includes the ability (at this level) to control whether quantities are considered in on-hand calculations.
   - All updates related to Warehouse Management Systems are assumed to be against on-hand supply.
4. When adjusting inventory value based on the processing of a return (a receipt) or an inventory adjustment (positive or negative), Selling and Fulfillment Foundation publishes the change to be used by a general ledger accounting application. For more information about the inventory transactions and the specific data published, see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide* for time triggered transaction appendix.

POSITIVE INVENTORY ADJUSTMENTS

5. If the average cost of the item is not known, it is taken into stock at $0.00 cost and manual cost adjustments are required.

NEGATIVE INVENTORY ADJUSTMENTS

6. If on-hand quantity results in a negative number, the inventory value is also reflected as a negative number. No change is made to the average cost due to this transaction.

7. If an adjustment was made from a zero stock position and the average cost of the item is not known, the inventory value remains at $0.00.

RETURNS

8. If the average cost of the item is not known, it is taken into stock at $0.00 cost and manual cost adjustments are required.

Returns and Inventory Adjustment Scenarios

Example 6–14 through Example 6–17 illustrate how Selling and Fulfillment Foundation processes returns and makes inventory adjustments.

Example 6–14 Returns - Scenario 1 - See Figure 6–7

A return is created in Selling and Fulfillment Foundation from an existing sales order. The customer ships the product back to the designated return node. In this instance, the product is returned to the same facility from which it was shipped. The return is received and moved into stock (as on-hand). Inventory value is recalculated using the current average cost for the item at the return node. The change in inventory value is published to the general ledger accounting application.

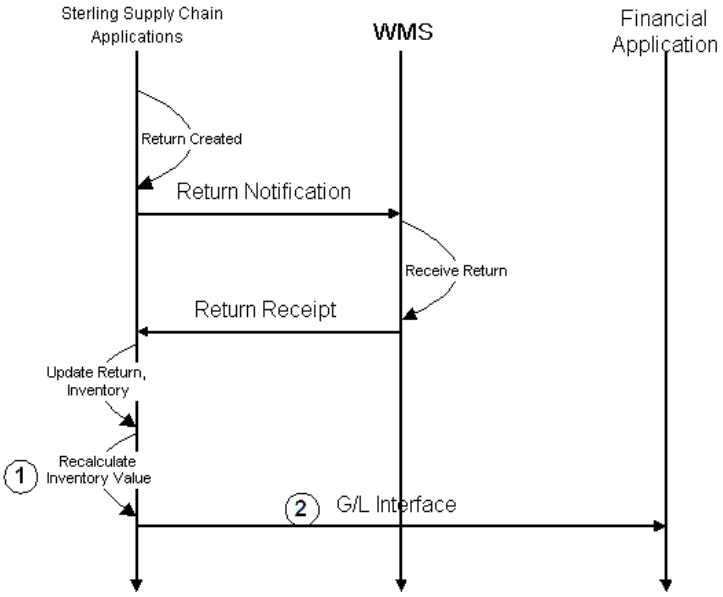
Example 6–15 Returns - Scenario 2 - See Figure 6–7

A return is created in Selling and Fulfillment Foundation from an existing sales order. The customer ships the product back to the designated return node. In this instance, the product is returned to a different facility than that from which it was shipped. The return is received and moved into stock at the return facility (as on-hand). Inventory value is recalculated using the current average cost for the item at the return node. The change in inventory value is published for the financial application.

For both Return scenarios 1 and 2 the general information flow is as follows:

1. Selling and Fulfillment Foundation recalculates the Inventory Value
   a. Current Average Cost = $4.75
   b. Current Inventory Value = $2500
   c. New Inventory Value = $2500 + (10 * $4.75) = $2547.50
2. Selling and Fulfillment Foundation publishes the information pertaining to the change in inventory value (used for integration to financial applications). For more information about the inventory transactions and the specific data published, see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide* for time triggered transaction appendix.

Figure 6–7 Return Scenario 1 & 2



Example 6–16 Inventory Adjustment - Scenario 1 - See Figure 6–8

After cycle counting, the inventory of item ABC at Node1 needs to be decreased by 4 units. This is reported in the Warehouse Management System and interfaced to Selling and Fulfillment Foundation. On completing the adjustment within the Sterling WMS, Selling and Fulfillment Foundation recalculates the inventory value to reflect the "loss" of 4 units of ABC at the average cost of ABC at Node1. The change in inventory value is published for the financial application.

1. Selling and Fulfillment Foundation recalculate the Inventory Value
   a. Current Average Cost = $11.20
   b. Current Inventory Value = $12,500
   c. New Inventory Value = $12,500 - (4 * $11.20) = $12,455.20
2. Selling and Fulfillment Foundation publishes the information pertaining to the change in inventory value (used for integration to financial applications). For more information about the inventory transactions and the specific data published, see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide* for time triggered transaction appendix.

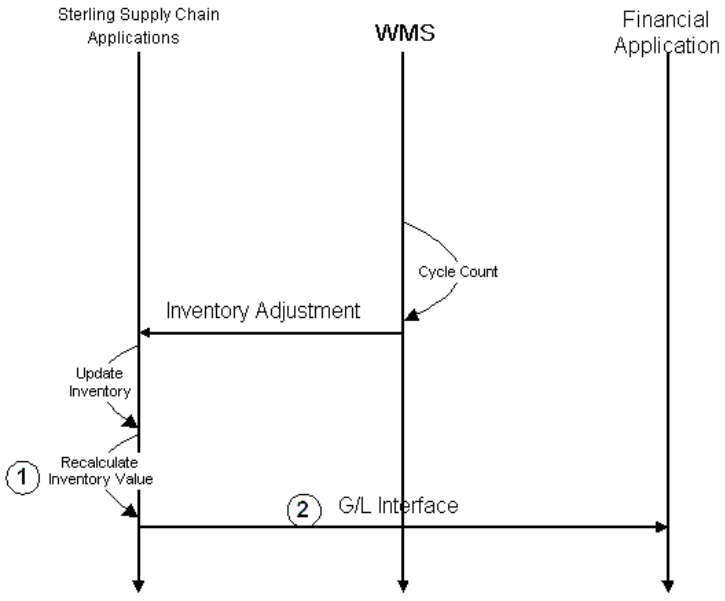
Example 6–17 Inventory Adjustment - Scenario 2 - See Figure 6–8

After cycle counting, the inventory of item DEF at Node1 needs to be increased by 13 units. This is reported in the Warehouse Management System and interfaced to Selling and Fulfillment Foundation. On completing the adjustment within the Sterling WMS, Selling and Fulfillment Foundation recalculates the inventory value to reflect the "gain" of 13 units of DEF at the average cost of DEF at Node1. The change in inventory value is published for the financial application.

1. Selling and Fulfillment Foundation recalculates the Inventory Value
   a. Current Average Cost = $20.00
   b. Current Inventory Value = $17,800
   c. New Inventory Value = $17,800 + (13*$20.00) = $18,060
2. Selling and Fulfillment Foundation publishes the information pertaining to the change in inventory value (used for integration to financial applications). For more information about the inventory transactions and the specific data published, see the *Selling and*

Fulfillment Foundation: Application Platform Configuration Guide for time triggered transaction appendix.

Figure 6–8 Inventory Adjustments - Scenario 1 & 2



6.9.1.3.4 Work In Process Handling

Work-In-Process handling describes the process of assembling kits within a warehouse. The process involves consuming component products (that can be sold individually) according to a parent bill of material, to produce a quantity of the parent item. Production of the parent item and reporting its "receipt" into inventory complete the task.

Selling and Fulfillment Foundation handles work-in-process as follows:

1. For kit processing within a Warehouse Management System, all "production" activity of the parent item is reported as positive inventory adjustments. All "consumption" activity of component items is reported as negative inventory adjustments.

2. For component items, on report of consumption, Selling and Fulfillment Foundation adjusts the inventory value at the production location.
3. For component items, on report of consumption, Selling and Fulfillment Foundation publishes the negative change in inventory value for the financial application. For more information about the inventory transactions and the specific data published, see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide* for time triggered transaction appendix.
4. For parent items, on report of production, Selling and Fulfillment Foundation calculates the increase in inventory value as described in Section 6.9. Selling and Fulfillment Foundation also recalculates the average cost for the item at the production location.
5. When a receipt is made against an existing on-hand balance that is less than zero, Selling and Fulfillment Foundation handles the receipt as described in Section 6.9.1.1.7.
6. For parent items, on report of production, Selling and Fulfillment Foundation publishes the change in inventory value for the financial application. For more information about the inventory transactions and the specific data published, see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide* for time triggered transaction appendix.

Work In Process Scenarios

Example 6–18 illustrates how Selling and Fulfillment Foundation processes the Work In Process orders.

Example 6–18 Work In Process Scenario - See Figure 6–9

The Warehouse Management System reports the production of 20 units of kitted item A12 at Node1. It also reports the consumption of the component inventory that went into the 20 units of A12 – 20 units of C1, 40 units of C2 and 60 units of C3.

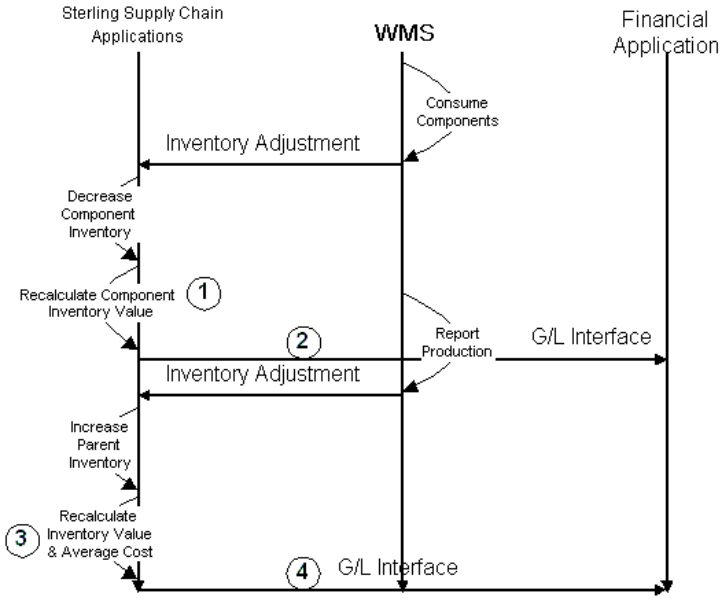
The standard makeup of A12 (one unit) is 1 unit of C1, 2 units of C2, and 3 units of C3. Assume that the following Cost Factor Group is associated with the production of work orders for this Enterprise:

Cost factor name	Calculation method	Application Method	Percentage	Value	Comment
Labor	Percentage	-	20%		20% of the base cost is added to arrive at the labor cost
Supplies	Amount	Quantity	-	3.00	$3.00 is added to every finished kitted unit as supplies cost

1. On decrease of component inventory within Selling and Fulfillment Foundation, the solution recalculates the inventory value for each component as follows:
   a. For C1, 20 *Average Cost of C1 ($1.50) = $30
   b. For C2, 40 * Average Cost of C2 ($0.50) = $20
   c. For C3, 60 * Average Cost of C3 ($1.75) = $105
2. Selling and Fulfillment Foundation publishes each inventory value change in a separate event for the financial application. For more information about the inventory transactions and the specific data published, see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide* for time triggered transaction appendix.
3. The value of production of 20 Units of A12 is as follows:
   a. Component Cost = $30 + $20 + $105 = $155
   b. Labor Cost (20%) = $31
   c. Supplies = $3/Unit = $60
   d. TOTAL COST OF A12 = 155+31+60 = $246
   e. Existing Inventory Value = $850, Quantity = 100
   f. New Inventory Value = $1096, Quantity = 120
   g. New Average Cost A12 = 1096/120 = $9.133333

4. Selling and Fulfillment Foundation publishes the information pertaining to the change in inventory value (used for integration to financial applications). For more information about the inventory transactions and the specific data published, see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide* for time triggered transaction appendix.

Figure 6–9 Work In Process Scenario



6.9.1.4 Inventory Costing for Drop-Ship Products

Selling and Fulfillment Foundation processes drop-ship orders as follows:

ORDER CREATE

1. Selling and Fulfillment Foundation determines the unit cost as described in Section 6.9.1.3.
2. Selling and Fulfillment Foundation transfers the unit cost on the parent sales order line to the unit price of the chained purchase order.

If the unit cost is not overridden on the sales order line, the standard cost factors are subtracted from the unit cost before Selling and Fulfillment Foundation transfers it to the unit price of the chained purchase order.

3. Selling and Fulfillment Foundation publishes the newly created chained PO for the financial application. This can also be used for vendor invoice reconciliation.
4. The unit price is locked for the chained PO.

ORDER CHANGE

5. Selling and Fulfillment Foundation transfers any update to the unit cost on the parent sales order line to the corresponding chained PO line.
6. Selling and Fulfillment Foundation does not propagate any update to the unit price on the chained PO unit price back to the unit cost on the parent sales order line.

SHIP CONFIRMATION

7. When Selling and Fulfillment Foundation receives a shipment confirmation notice, an "on success" event is published. This event can be used to simulate a receipt in the financial application. Also, the same event can be used to post entries into A/P and COGS. This is only done for shipments that are direct shipped.

INVOICE CREATION

8. Selling and Fulfillment Foundation creates an invoice for the sales order and this invoice is posted to A/R and sales as usual. All transactions have a shipment number reference that is used to tie them together.

Drop-Ship Order Scenarios

Example 6–19 illustrates how Selling and Fulfillment Foundation processes drop-ship orders.

Example 6–19 Drop-Ship Order Scenario - See Figure 6–10

A customer places an order for a quantity of 2 pieces of item DEF. The order is priced according to pricing rules for the customer. The product is sourced from Supplier1 and shipped directly to the customer. The

expected cost, based on the primary supplier of DEF is retrieved from the replacement cost for DEF. The order is scheduled and released. This generates a "chained" PO for Supplier1 to ship 2 pieces of DEF to the customer. Supplier1 ships DEF to the customer and sends an invoice to the Enterprise. The recording of the invoice triggers a shipment confirmation against the "chained" PO which in turn propagates a shipment confirmation against the original sales order.

When the shipment confirmation event is issued against the chained order, a receipt is simulated in the financial application. Also, A/P and COGS general ledger entries are made at this point. These entries are made based on information on the chained PO line. An invoice is created for the original sales order and is published for the financial applications. This is posted into A/R and Sales.

1. Order Creation
   a. The Order is priced according to the price list defined by the seller
   b. The unit cost is stamped on the order line
2. Chained PO Creation

   If the unit cost has been overridden (specifically set on the sales order line), the unit cost from the sales order line becomes the unit price on the chained PO. At this point, the unit price is locked for the chained PO. If the unit cost has not been overridden, the Order is priced according to the price list defined by the seller. For logical kit items, the unit cost from the sales order is not propagated to the PO. From this point until shipment confirmation of the PO, the PO unit price can be maintained either:

   - On the PO, which does not propagate back to the Sales Order unit cost
   - On the sales order maintenance of the unit cost. This propagates to the PO unit price.
3. Shipment Confirmation
   a. On receipt of shipment confirmation from the financial application (triggered by receipt of A/P invoice from supplier), Selling and Fulfillment Foundation records the transaction against the chained PO.
      - The chained PO pipeline has no invoice transaction in it. It is not required in this scenario.

b. Shipment confirmation against the chained PO triggers, through internal Selling and Fulfillment Foundation mechanisms, a shipment confirmation event against the original sales order.

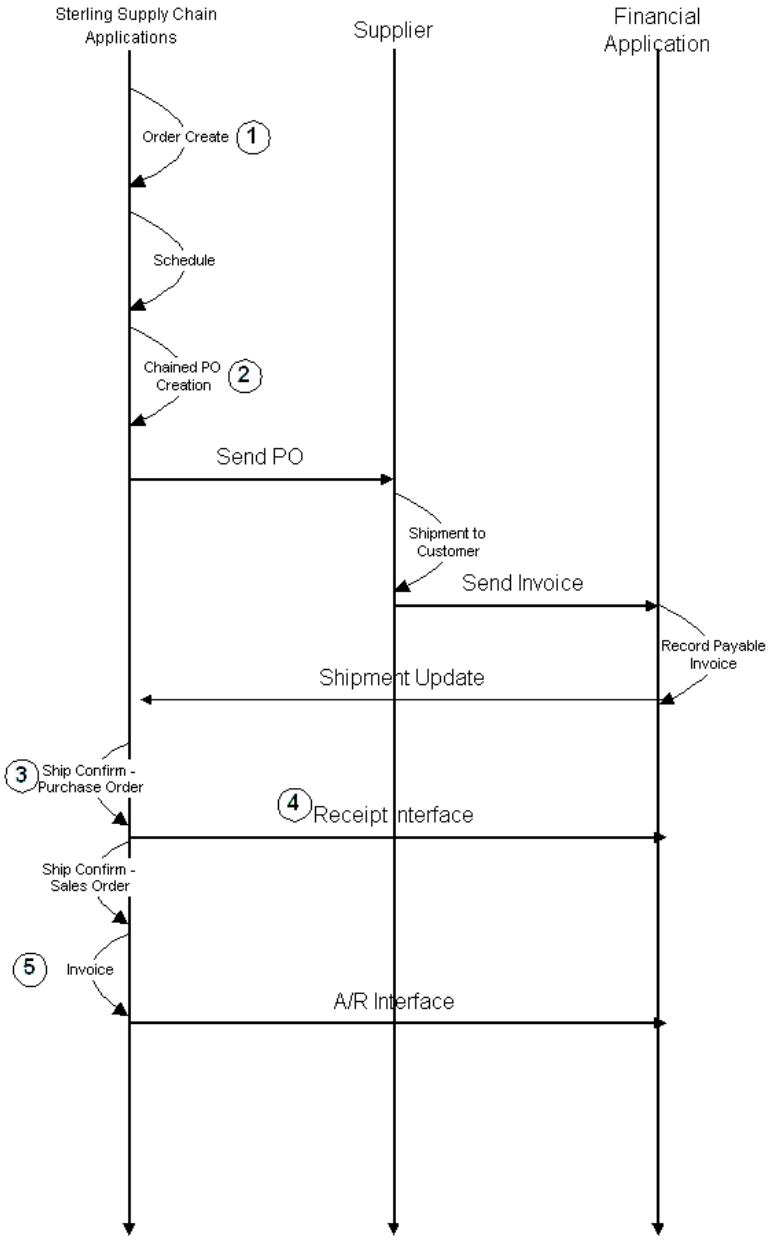
4. Receipt Interface

   Though no inventory has been received at the enterprise, a receipt is used to match against the supplier invoice. It contains the PO line detail to satisfy financial application integration.

5. Invoice Creation

   For the sales order line, Selling and Fulfillment Foundation creates an invoice line that is used for creating A/R entries

Figure 6–10 Drop-Ship Scenario



6.9.2 First In First Out (FIFO)

For the FIFO costing method, the Selling and Fulfillment Foundation agents match receipts and shipments in first in, first out (FIFO) order and record the matched cost information. This matched information is used for reporting and posting to financial systems. Agents also post inventory cost data to financial and inventory systems.

As returns are processed, decrements to inventory are rolled back in FIFO order, meaning that the last item that was shipped is the one assumed to be returned in order to determine the cost of the returned item.

Selling and Fulfillment Foundation provides APIs to perform receipts of inventory, shipments, and returns. A user exit is called for adjustments, shipments, and returns in order to provide the logic for general ledger account number determination and cost determination, if needed. Inventory adjustments accumulate for manual review by a cost accountant before posting records to the matching tables. This allows cost accountants to adjust cost at the aggregate item level and cancel out adjustments.

6.10 Hot SKU

A Hot SKU is a popular item with a high volume of requests during a specific period of time. For example, the release of a new CD from a very popular artist might generate a high volume of requests for that item right as it goes on the market.

Under high transaction load, the presence of Hot SKUs can create a situation where multiple lock requests are placed on the same item at the same time. When that happens, one transaction holds the lock and the rest are blocked.

Selling and Fulfillment Foundation can identify a certain item as a Hot SKU automatically. Under normal conditions, when Hot SKUs are not detected, the system would place a lock on an item during inventory changes, update the supply and demand tables, and then release the lock. When an item is considered hot, the system does not lock it. Instead, the changes are inserted into two additional tables, one for demand and one for supply.

Selling and Fulfillment Foundation always looks at both the base and additional tables. In order to keep the additional tables from growing

infinitely, Selling and Fulfillment Foundation provides an agent that reconciles the base and additional supply and demand tables.

For detailed information regarding the Hot SKU functionality, refer to the *Selling and Fulfillment Foundation: Performance Management Guide*.

6.11 Count

Accuracy of inventory level is critical to a supply chain. Inventory levels are the key to having better customer satisfaction and demand planning.

A count system allows you to carry out counts in a planned or in an ad hoc manner. A common type of count employed is year-end inventories. While this is exhaustive, it is also time consuming, and does not ensure accuracy throughout the year. The other method is to only count items based on velocity or price every quarter. However, the best method is to ensure count performed periodically in the system.

Selling and Fulfillment Foundation allows enterprises to define count programs by item category and define the number of times a category of item needs be counted in a specific period at a specific node, nodes in a region, or all participating nodes.

Selling and Fulfillment Foundation allows enterprises to initiate count requests through console on an ad hoc basis for a specific node, all nodes in a specific region, or all participating nodes.

Examples of count requests that can be created by an enterprise include:

1. Request to count an item, product class, and UOM combination
2. Request to count an item category
3. Request to count items that have unit price in a specific range

Counts for count requests created in an ad hoc or planned fashion are performed at participating nodes. The resolution of variances found, if any, ensure inventory correctness at that node. The Enterprise initiating a request is kept informed when all the nodes that have been requested to count, have finished counting.

For more details on count execution at a node, see the *Sterling Warehouse Management System: Concepts Guide*.

6.12 Inventory Consignment

Inventory Consignment is a concept in which inventory availability of one inventory organization can be accounted as part of inventory availability of another inventory organization.The inventory organizations participating in the inventory consignment program are called consumable inventory organizations and consuming inventory organizations.

A consuming inventory organization can choose a list of consumable inventory organizations to participate with itself. Inventory availability of consumable inventory organizations can be accounted as part of inventory availability of the consuming inventory organization. A consumable inventory organization cannot be a consuming inventory organization of another inventory organization. This means that only one level of relationship is allowed.

Inventory Consignment enables consuming inventory organizations to maintain an inventory at a consumable inventory organization until just prior to shipping, at which point ownership of the inventory is transferred from the consumable inventory organization to the consuming inventory organization. This enables a consuming inventory organization to greatly reduce its carrying cost, while still maintaining an accurate inventory picture because while it does not technically own the inventory until it is shipped, it can still be counted as an available inventory.

In an enterprise and vendor model, a enterprise's inventory organization can be configured as a consuming inventory organization. Inventory organizations of vendors, who participate with the enterprise, can be configured as a consumable inventory organizations.

Note: The catalog organization can be different between inventory organizations, thus the same product can be modeled as different items in different catalogs. In such a situation, the system uses GlobalItemID (GTIN) to reference to these items.

When transferring inventory ownership, the system assumes that the consumable inventory organization and consuming inventory organization have the same product class.

While obtaining availability of a consumable inventory organization through promising functions, system will try to maximize its availability by assigning the excess demand of the consuming inventory organization to other participating consumable inventory organizations first, if possible.

Inventory rules are read based on the consuming inventory organization. In a configuration where a consumable inventory organization participates with multiple consuming inventory organizations, the system validates whether all the participating consuming inventory organizations have the same set of inventory rules. These inventory rules are Rules within ATP rules such as Use Item Based Allocation, Use Future Supply of Inventory Node Control, Future Supply Type Safety Factor Percentage, Onhand Supply Type Safety Factor Percentage, and Default Distribution Rule ID.

The inventory rules of the consuming inventory organization will be used while obtaining inventory availability for the consuming inventory organization and consumable inventory organization. The sourcing and scheduling related rules of the inquiring organization will be used for the promising functions.

7

Order Management

Order Management builds upon the concepts detailed in Chapter 4, "Process Modeling Concepts". Order Management uses the process type pipelines developed in Process Modeling to send an order document through its various stages such as creation, scheduling, releasing, and shipment, and track its lifecycle.

The Order console provides access to order information. Managers and customer service representatives can view order information in real-time to handle alerts and correct problems, ensuring on-time execution of orders.

7.1 Parts of an Order

In Selling and Fulfillment Foundation, an order can be broken down into an order header level, order line level, and order release level.

An order is made up of order lines. An order line is a line on the order containing information relative to a specific item being ordered.

For example, a Buyer creates an order for three computer monitors, three keyboards, and three printers. When looking at the entire order you are at the order header level. The monitors, keyboards, and printers are the order lines with a quantity of 3 each specified.

Some attributes pertain to both the order header level and order line level. For example:

- Ship to information, such as the street address where the order is sent
- Personalization instructions, such as gift wrapping
- Shipping and handling costs

- Price Promotions

Some attributes only pertain to the order header level. For example:

- Bill to information, such as the street address where the invoice is sent
- Payment information, such as whether the Buyer paid by credit card or check
- Additional attributes, such as order identification, order creation, shipping, and financial information
- Exchange type, if the order is an exchange order. It can be a regular exchange, advanced exchange, or a pre-paid exchange.

Some attributes only pertain to the order line level. For example:

- Item identifier
- Unit of measure, such as a single unit or a dozen
- Quantity of the line item ordered
- Order line status, such as Created or Shipped
- Line schedules
- Inventory attributes, such as lot number and revision number
- Serial number
- Pricing information, such as the unit price for the line item
- Mark for address
- Gift attributes, if the order line is a gift

7.2 Classifications

You can define product item classifications and classification hierarchies that can be used within Selling and Fulfillment Foundation for actions such as sourcing, associating services, determining shipping preferences, obtaining a list of items for ordering, and so on. By defining a classification you identify an item attribute as having a specified use in Selling and Fulfillment Foundation.

You can also define hierarchical item groupings for a classification. These groupings can be used to further refine the items affected by classification purpose.

7.3 Order Pipelines

When modeling your business process, you can create pipelines for order, negotiation, planned order, return, and purchase order process types. A pipeline consists of the different statuses a document goes through during fulfillment. You can also set up transactions consisting of events, actions, and conditions, as they pertain to the pipeline you are configuring.

An order document could travel through the following process-type pipelines:

- Order fulfillment
- Negotiation
- Purchase Order
- Master Order
- Return
- Quote fulfillment

7.3.1 Order Fulfillment Pipeline

Beginning from its creation, an order flows through a set of transactions and statuses until its completion. This chain of transactions and order statuses is called the Order Fulfillment pipeline.

An Order Fulfillment pipeline provides Selling and Fulfillment Foundation with a means to perform actions, such as sending notifications or logging alerts, dependent on the location of the order in the pipeline. It also provides you with a means to track an order from creation to completion and perform any necessary manual interventions. The figure below illustrates the structure of an Order Fulfillment pipeline.

The Order Fulfillment pipeline from which your business runs is unique to how your system administrator has set up your business environment. However, every order pipeline generally begins with a transaction that

creates an order and ends with a transaction that indicates an order has been shipped or, if applicable, returned.

A transaction is an occurrence that needs to be tracked and can call certain actions to be performed. An order status describes what state an order is in and moves it from transaction to transaction.

The following statuses may be used in an Order Fulfillment pipeline, depending how it is configured within your system:

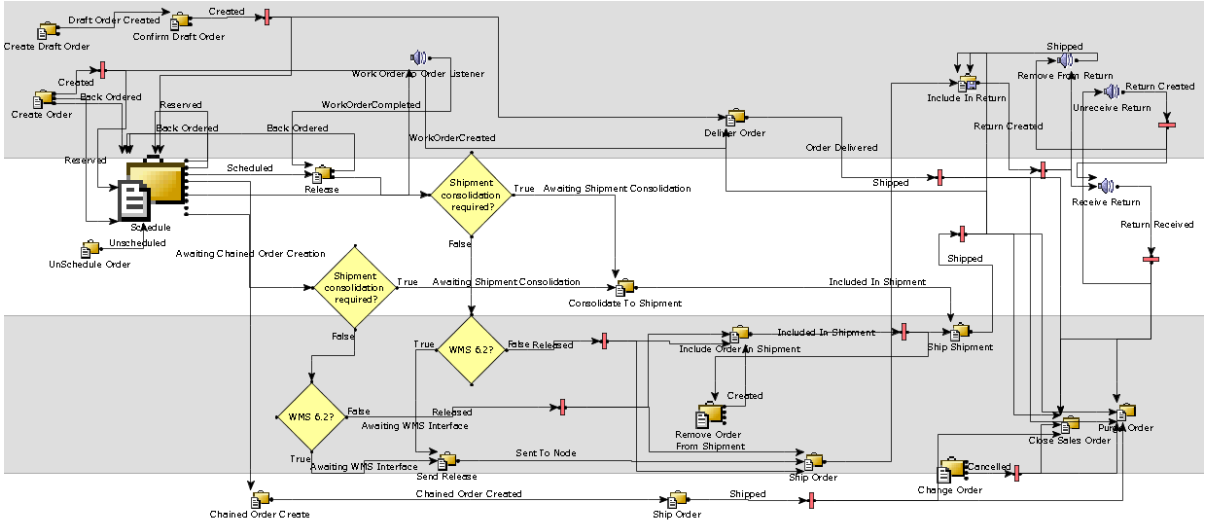
- Accepted - The negotiated terms have been accepted by both parties and the order is ready to be released.
- Awaiting Chained Order Creation - The order is to move into chained order created status. Section 7.6.1, "Chained Orders".
- Awaiting Shipment Consolidation - The order is comprised of multiple shipments that are consolidated at a node before further shipment or delivery is made.
- Backordered - The order has been created, but there is not enough inventory to schedule the order. The order remains backordered until inventory is available.
- Backordered From Node - The order has been created and released to the node, but the node does not have enough inventory to fulfill the order. At this point, if configured, inventory is considered unavailable for order promising until a physical count is performed at the node.

  If configured, internal inventory within a Selling and Fulfillment Foundation node can be promised against a future supply of inventory.
- Being Negotiated - The order is being negotiated in the negotiation pipeline.
- Cancelled - The order has been canceled.
- Chained Order Created - A chained order has been created and sent to the applicable node.
- Created - The order has been created.
- Delivered - The service and product lines included on the work order have been delivered.

- Draft Order Created - A draft order has been created in the Create Order console, Order Entry screen. All aspects of this order can be modified until it is confirmed.
- Draft Order Reserved - A draft order has been created and inventory is reserved for the order. If this order is not confirmed and the order is removed or canceled, the inventory reserved for this order can be used for other orders.
- Included In Shipment - The order is included in a shipment.
- Order Invoice Created - An invoice has been created for the order.
- Procurement Purchase Order Created - A procurement purchase order has been created in the Create Order Console, Order Entry screen.
- Procurement Purchase Order Shipped - A procurement purchase order has been shipped.
- Procurement Transfer Order Created - A procurement transfer order has been created in the Create Order Console, Order Entry screen.
- Procurement Transfer Order Shipped - A procurement transfer order has been shipped.
- Ready To Negotiate - If negotiation is performed between the Enterprise and the ship node, the order is sent to a separate negotiation pipeline.
- Received - The order has been received by the buyer.
- Received As Components - If an item in the order consists of kit components, this status indicates that the Buyer has received all components.
- Released - There is enough inventory to schedule the order for fulfillment. The order is released to the Application Consoles, the Selling and Fulfillment Foundation Warehouse Management System, or another third-party warehouse management system.
- Reserved - The order has been created, but it is not ready to schedule for shipment yet.
- Return Created - The buyer is returning one or more items included in the order.
- Return Received - Returned items have been received at the return node.

- Scheduled - The applicable node(s) have the inventory to fulfill the order and it can be scheduled for release.
- Sent to Node - The order has been sent to the node, as an order release.
- Shipped - The order has been shipped.
- Shipment Delayed - All or part of the order shipment has been delayed.
- Shorted - The order contains less quantity than requested. The order is closed.
- Unreceived - The order has not been received by the buyer.
- Unscheduled - The order has been removed from Scheduled status and any inventory that has been reserved for the order at the scheduled node(s) is canceled.

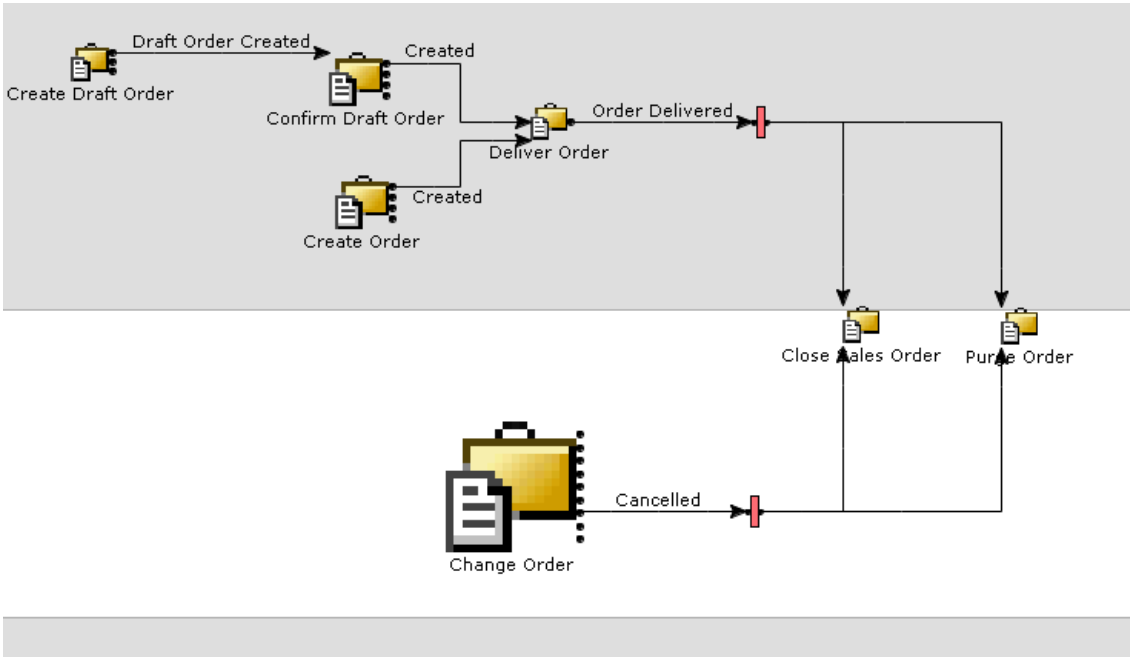
Figure 7–1 Order Fulfillment Pipeline



Note: The default pipeline and statuses can be configured according to your business practices. For more information about configuring order fulfillment pipelines and statuses, see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide*.

The following is a graphical representation of the default Sales Order Service Fulfillment pipeline as it appears in the Applications Manager.

Figure 7–2 Sales Order Service Fulfillment Pipeline



7.3.2 Negotiation Pipeline

Negotiation is the process in which two organizations can settle the conditions of a process-type document. You can configure negotiation after an order is created.

Whenever a negotiation process needs to be incorporated in an order fulfillment pipeline, a negotiation document is created from the fulfillment document (for example, order or planned order). All negotiations are conducted over the negotiation document and the results are applied to the original document.

A ***negotiation pipeline*** allows participants to negotiate on details of a current transaction. This pipeline can be configured to occur anywhere before release within an existing pipeline. The result from the negotiation process is either in the form of acceptance with the potential to update some of the negotiated values or in rejection of the terms of the transaction by one or both of the participants.

7.3.2.1 Negotiation Responses

There are two organizations involved in a negotiation process. The initiator organization and the negotiator organization. Negotiation is carried over by sending responses to each other. A response can be for one order line or all of the order lines. Once a response of one organization is accepted by the other organization, the negotiation moves to completed status. After negotiation is completed, the negotiated terms are applied to the original document.

A response is identified by a response number. Every response needs to have a "for response number" which is the response number of the last response from the other organization. Table 7–1, "Negotiation Responses" describes the negotiation responses.

Table 7–1 Negotiation Responses

Response Number	Response Name	Description
1100	Offer	The response is an offer from the initiator. Only the initiator organization can send this response.
1200	Counter Offer	The response is a counter offer from the negotiator. Only the negotiator organization can send this response.

Table 7–1 Negotiation Responses

Response Number	Response Name	Description
1300	Reject	The response is a rejection from the negotiator. Only the negotiator organization can send this response.
1400	Remove	The initiator wants to remove the line from negotiation. Only the initiator can send this response. This response is available only at the line level. Once a line is removed, it is assumed that the line has been negotiated and no further negotiation is allowed on that line.
1500	Accept	The sending organization accepts the terms of the other organization. Both the initiator and negotiator can send this response. Once a header or line is accepted, it is assumed that the header or line has been negotiated and no further negotiation is allowed on that header or line.

7.3.2.2 Negotiation Actions

If the response action is Accept or Remove, it should have an associated action. The action for an order and planned order is CANCEL. The action information is used when negotiated terms are applied on the original document. The two organizations may agree on a lesser quantity than what was on the original document. In that case, the specified negotiation action is taken on the remaining quantity.

A negotiation can have a header and multiple lines. The negotiation status is maintained at the header level and is derived from the last response on the negotiation.

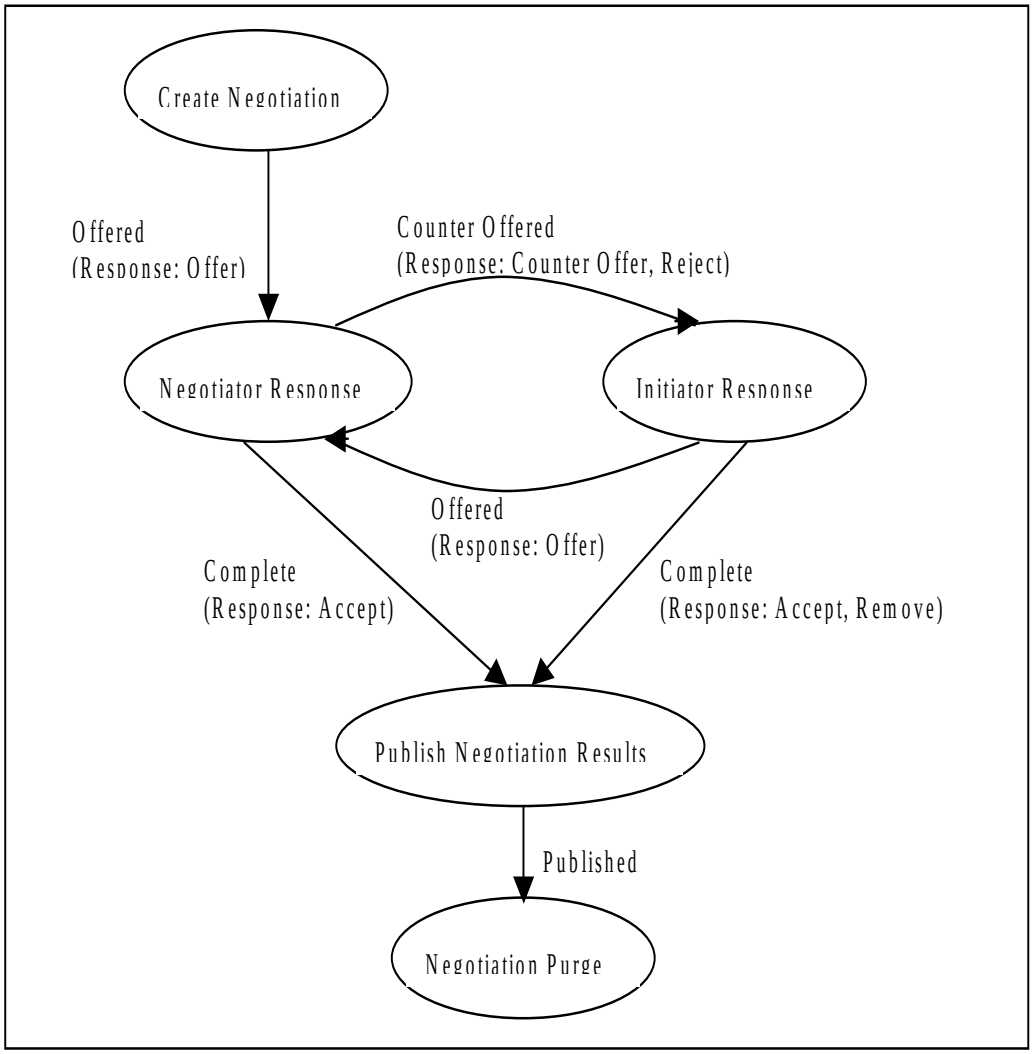
Table 7–2 Negotiation Statuses

Status Code	Status Name	Description
1000	Offered	The last response on the negotiation was from the initiator organization. The initiator can send another offer before the negotiator organization responds to the original offer.
1100	Counter Offered	The last response on the negotiation was from the negotiator organization. The negotiator can send another counter-offer before the initiator organization responds to the original counter-offer.
1200	Completed	The header and line terms on all of the lines have been negotiated by the two organizations.
1300	Published	The negotiated terms have been published and applied to the original document.

7.3.2.3 Negotiation Process

Figure 7–3, "Negotiation Process" represents the transactions and statuses involved in a typical negotiation process. Negotiation statuses are carried at the header level for the entire document being negotiated.

Figure 7–3 Negotiation Process



Negotiation can be configured after order creation.

7.3.3 Quote Fulfillment Pipeline

A quote is a legally binding agreement between a seller and a buyer to purchase a pre-determined set of items and quantities, at a pre-determined price, to be delivered by a specific date. A quote can be accepted or rejected by a buyer, or it can expire on a specified date if an order has not been placed.

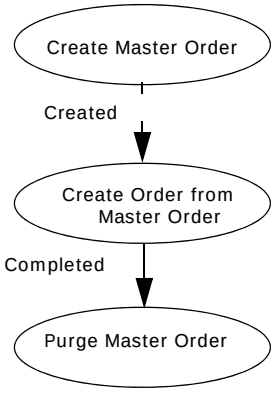
For information about the Quote Fulfillment pipeline, refer to *Selling and Fulfillment Foundation: Setting Up Quotes*.

7.3.4 Master Order Pipeline

A ***master order*** is a document indicating expected recurring sales of multiple products over a period of time. A master order is typically created for a customer to purchase a series of products for shipping.

Figure 7–4, "Master Order Pipeline" represents a typical master order process-type pipeline.

Figure 7–4 Master Order Pipeline



7.4 Exchange Orders

An exchange order is a sales order with the ORDER_PURPOSE flag set to EXCHANGE. It can only be based off a return order, and behaves like a sales order.

An exchange is even when the value of the products being sent to the customer as an exchange is equivalent to the value of the products the customer returned. For example, a customer returns a blue shirt and wants the same style shirt and size in exchange, only red.

An exchange is uneven when the value of the products being sent to the customer as an exchange is greater or smaller than the value of the products the customer returned. For example, a customer returns a blue shirt and wants a red shirt as well as a green shirt in return. In that case, new payment processing needs to be done.

There are three types of exchanges:

Regular Exchange

A regular exchange occurs when the enterprise waits to receive the return from the customer before sending the exchange. The exchange item is only sent when the return item has been received, and if necessary, when any remaining balance has been paid off.

For example, a customer sends back a red shirt asking for a blue shirt in exchange, and wants to purchase an additional green shirt. Once the enterprise receives the red shirt, it charges the customer for the green shirt, and sends both of them at once.

Advanced Exchange

An advanced exchange occurs when the enterprise ships the exchange item as soon as the exchange is created, without waiting to receive the return item. For example, a cell phone company may want to send a new cell phone to its customer as soon as the CSR receives the request. The company assumes that it receives the return, and does not wait to issue the exchange.

Advanced Pre-Paid Exchange

An advanced pre-paid exchange occurs when the enterprise charges the customer for the exchange item, ships it, and then refunds the customer upon receipt of the return item. For example, if a customer wants to

exchange a red shirt for a blue shirt, the enterprise charges the customer for the blue shirt, ships it, and upon receipt of the red shirt, refunds the value of the red shirt to him.

In cases that involve a provided service on the exchange order, Selling and Fulfillment Foundation synchronizes exchange orders with return orders to enable the creation of a single work order that contains uninstallation and pickup of the old item along with delivery and installation of the replacement item.

7.5 Bundles

A bundle is a package containing both products and services. It is composed of a bundle parent and one or more child lines. The child lines can be either products, services, or other bundles. The bundle parent and any child line bundles are created for marketing purposes only, it cannot be fulfilled as part of an order. An example of a bundle order would be a bedroom package offered by many furniture companies. The bedroom package contains many different products and services like a dresser, a nightstand, a bed set bundle containing a mattress and bed frame, and a service for the installation of the bedroom package. Figure 7–5 illustrates a typical bundle.

Figure 7–5 Example of a Bundle - Bedroom Package

To model bundles in Selling and Fulfillment Foundation, the order structure supports a hierarchical order line. Table 7–3, "Bundle Modeling in the Selling and Fulfillment Foundation" describes bundle modeling in Selling and Fulfillment Foundation.

Table 7–3 Bundle Modeling in the Selling and Fulfillment Foundation

Order Line	Bundle Parent Line
Bedroom Package (Parent Line)	**Blank**
Dresser	Bedroom Package
Nightstand	Bedroom Package
Bed Installation	Bedroom Package
Bed Set Package (Parent Line)	Bedroom Package
Mattress	Bed Set Package

Table 7–3 Bundle Modeling in the Selling and Fulfillment Foundation

Order Line	Bundle Parent Line
Bedroom Package (Parent Line)	**Blank**
Bed Frame	Bed Set Package
Headboard	Bed Set Package
Footboard	Bed Set Package

As Table 7–4 illustrates, each component indicates its parent line and each parent line indicates whether or not it is a parent bundle line. If an order line has a bundle parent line, the line is treated as part of a bundle and referred to as a bundle component.

Selling and Fulfillment Foundation provides the following additional capabilities around bundles:

- Creating an Order
- Ordered Quantity Computation for Bundle Components
- Inventory Updates
- Service Association
- Order Line Schedule Attributes Synchronizing
- Order Modification
- Order Scheduling
- Pricing
- Chained Orders
- Derived Orders
- Returns

7.5.1 Creating an Order

You can create an order containing n-tier bundles. The order creation process reads the bundle definition from the catalog and honors the sourcing constraints for reserving the product as long as the bundle parent information is passed. If you maintain a catalog using an external

system, Selling and Fulfillment Foundation reads the bundle definition by passing the bundle parent information to the external system.

The Sterling Distributed Order Management provides the ability for a bundle to create subordinate bundles in order to fulfill communication with third parties or to perform some other related unit of work. This ensures that the subordinate bundles report status updates, where appropriate, back to the parent bundle and that the parent bundle can communicate change in linked values to each subordinate bundle. The example given in Table 7–3, "Bundle Modeling in the Selling and Fulfillment Foundation" is of n-tier bundles.

Using Figure 7–5, "Example of a Bundle - Bedroom Package", the following examples provide details about the information that will be passed based on a customer's requirements, assuming the bundle described in Table 7–3 is present in the catalog:

Example 7–1 Customer Requests the Standard Bedroom Package

In this example, the customer requests the standard Bedroom Package. Since there is no change to the Bedroom Package bundle, only the Bedroom Package information is passed when creating the order. The following order lines are created for the Bedroom Package bundle:

- Bedroom Package (Parent Bundle)
- Dresser
- Nightstand
- Bed Installation
- Bed Set Package (Parent Bundle)
    - Mattress
    - Bed Frame
    - Headboard
    - Footboard

Example 7–2 Customer Requests a Modified Version of the Bedroom Package

In this example, the customer wants the standard Bedroom Package but does not want a nightstand. Since the requested components for the Bedroom Package has changed, the Bedroom Package bundle and its

immediate components are passed. The following order lines are created for the modified Bedroom Package bundle:

- Bedroom Package (Parent Bundle)
- Dresser
- Bed Installation
- Bed Set Package (Parent Bundle)
    - Mattress
    - Bed Frame
    - Headboard
    - Footboard

Example 7–3 Customer Requests a Modified Version of the Bedroom Package and the Bed Set Package

In this example, the customer wants the standard Bedroom Package but does not want a nightstand or a mattress. Since both the Bedroom Package bundle and the Bed Set bundle has changed, both the Bedroom Package and its components and the Bed Set and its components are passed. The following order lines are created for the Bedroom Package bundle:

- Bedroom Package (Parent Bundle)
- Dresser
- Bed Installation
- Bed Set Package (Parent Bundle)
    - Bed Frame
    - Headboard
    - Footboard

7.5.2 Ordered Quantity Computation for Bundle Components

A ordered quantity of a bundle component can be independent of its parent line or may be dependant on its parent. This is determined by kit quantity on the order line or the ratio quantity of the component order

line. If kit quantity is specified for the order line, then the quantity of the component line is the ordered quantity of the parent line multiplied by the kit quantity on the component line. When the kit quantity is specified, any change to the quantity of the parent line affects the quantity of the component and vice versa. The status change from the component to the parent and the parent to the component may also get propagated when the component maintains the kit quantity.

The component also can be created with fixed ordered quantity. In this case, not all changes to the quantity of the parent are applied to the components. The same is true with the quantity charge of the component. See Section 7.5.6, "Order Modification" for more information about how different modifications are handled.

Note: It is recommended that you create bundle components with kit quantity to facilitate quantity-related modification propagation from parent to component and component to parent.

7.5.3 Inventory Updates

The bundle parent line is a non-inventory item. No inventory updates take place for the bundle parent line.

7.5.4 Service Association

When bundles are read from the catalog, any association with service items may get established automatically.

Each product component in a bundle is associated with services applicable to the item that exists within the bundle.

Using Figure 7–5 as an example, if the dresser, the nightstand, and the bed set package have bed installation as an associated service, then the following associations are created on the order:

Table 7–4 Example of a Service Association in a Bundle

Product	Associated Service
Dresser	Bed Installation
Nightstand	Bed Installation

Product	Associated Service
Bed Set Package	Bed Installation

A bundle cannot have a delivery service as a component. This is to prevent multiple deliveries for the components of a bundle. If a bundle or bundle component needs to be delivered, a delivery service line can be passed to the order.

Delivery services can be associated with a bundle parent. If a delivery service that is associated with the bundle parent is passed to the order, all the components are delivered using one delivery service. The delivery service line can not be passed at a bundle component level. If a bundle component needs to be delivered using a specific delivery service, the service can be associated with the component.

7.5.5 Order Line Schedule Attributes Synchronizing

The bundle parent order line is not fulfilled, therefore the attributes ship node, expected shipment date, and expected delivery date are not computed for the parent. When these attributes change for the bundle components, the order line schedule attributes are determined for the bundle parent line.

For example, an order is created for the bundle as described in Table 7–3. When the schedule transaction runs, it determines the ship node, the expected shipment date, and the expected delivery date for the bundle parent.

7.5.6 Order Modification

There are some modifications that affect the parent if it is performed at the component level and there are some modifications that affect the parent if it is performed at the component level. All modifications that affect both the parent line and the child lines are explained below.

Adding Quantity

When quantity is added to a bundle parent, all components in the bundle that have kit quantity specified, have quantity added to them. The formula used for this is the change in the quantity of the parent

multiplied by the kit quantity of the component. Using Table 7–3 as an example, a customer has ordered one bedroom package and now wishes to purchase a second bedroom package. Changing the quantity for the bedroom package at the order line increases the quantity for all the components, assuming that the components are in ratio.

If the quantity is added at the component level, the quantity of the bundle parent is changed. The quantity of the parent line is the maximum of all the components. If this results in an increase in the quantity of the parent line then this modification is not allowed. The quantity modification at the component level cannot increase the quantity of the parent. Increasing the quantity for a component is allowed only if it was canceled earlier and the customer wishes to increase it again.

Splitting a Line

When splitting the quantity of a component, the kit quantity is propagated to the new line based upon a template. If the new line is a fixed quantity line, then it is not affected by modifications done to the parent line. The splitting of the entire bundle is not supported. If a ratio is maintained, the split line and the original line are considered one line for quantity modification purposes when the kit quantity is the same for both lines.

Adding a Line

A component can be added to an existing bundle parent line. The component can be a product, a service, or another bundle line.

Deleting a Line

A component line can be removed from a bundle. If a component is removed, the quantity of the parent bundle changes. When a bundle parent is removed all child lines are also removed.

Cancelling a Bundle

For any component with a set kit quantity is set, if the quantity is reduced at the parent line, the components are also reduced by an amount equal to the change in the quantity of the parent line multiplied by the kit ratio for the component. Cancelling the parent completely also cancels fixed quantity components.

If ‘Kit Quantity’ is not set for a component of the bundle, the quantity is not reduced from the component if the reduction occurs at the parent line.

The component quantity can be reduced independently, too. Cancelling a component causes the quantity of the parent line to be adjusted.

7.5.7 Order Scheduling

Each bundle parent indicates how a bundle is scheduled. The possible schedule values for a bundle are:

- Ship Together
- Ship Independently
- Deliver Together

These values are obtained from the catalog. If the catalog is not available, Selling and Fulfillment Foundation determines that the bundles ship together. These values only apply to product line components (service line components are ignored) and apply to components of the bundle at all levels.

7.5.8 Pricing

The bundle parent is treated like any other order line as far as pricing is concerned. The line total is computed like any other order. The order total includes the bundle parent as well as any components. A bundle can have a unit price, charges, and taxes like any other line.

There is an attribute at the order line level which can suppress the price of a line to be excluded from the order total or the overall amount a customer has to pay. If you want to have a price for a line but do not want to include it in the order total, set the attribute for the component. For bundles, if price of a component is included in the package itself but you want to store the price of an individual line in case of cancellation or a return, you may set the attribute on the line. In order to reflect the special price of a bundle, you may choose to set attributes for the components so that the price of the component is not reflected in the total price of the order.

Using Table 7–5 as an example, the pricing for the bedroom package bundle and its components could be:

Table 7–5 Bedroom Package Pricing

Bedroom Package	Line Total Amount	Price For Information Only
Bedroom Package (Bundle)	$500	
Dresser	$200	Y
Nightstand	$100	Y
Bed Set Package (Bundle)	$300	Y

Under normal circumstances, the order total would be the sum of all lines, which in this case is $1100. Instead, if the order should be priced differently based on the bundle parent, and you do not want the component level price to be included in the invoice, you can set an attribute at the component level during order creation. For more information about these attributes, see the *Selling and Fulfillment Foundation: Javadocs*.

7.5.9 Chained Orders

When a scheduled transaction searches for components and creates a drop-ship order, the bundle parent is pulled into the chained order for visibility purposes. The bundle parent is not linked to the parent order in the chained order. This means that any changes to the bundle parent in the parent order do not show in the chained order. Only scheduled attributes and quantity modifications are shown.

For more information about chained orders in general, see Section 7.6.1, "Chained Orders".

7.5.10 Derived Orders

When a derived order is created, if the order line is a bundle parent, all eligible components that are in ratio are also pulled into the derived order. The relationship of the bundle parent to the component is maintained in the new order. Any service lines included in the parent bundle are not included in the derived order.

For more information in general about derived orders, see Section 7.6.2, "Derived Orders".

7.5.11 Returns

A return can be created for components as well as for the bundle parent. When a return is created for a bundle parent, all returnable components which are in ratio are included in the return order. If a return is created for a component, the bundle parent is not included in the return order even if all of the components are included in the return order. For example, in Figure 7–5, "Example of a Bundle - Bedroom Package" if only the bed frame is returned, the bundle parent is not included in the return order. If the bedroom package is returned, all the child lines (dresser, nightstand, bed set package) are included in the return order.

For more information about returns in general, see Chapter 14, "Reverse Logistics".

7.6 N-Tier Orders

The Selling and Fulfillment Foundation Order Management provides the ability for an order to create subordinate orders in order to fulfill communication with third parties or to perform some other related unit of work. This ensures that the subordinate orders report status updates, where appropriate, back to the parent order and that the parent order can communicate change in linked values to each subordinate order.

Selling and Fulfillment Foundation provides the following types of n-tier orders:

- Chained Orders
- Derived Orders

Each n-tier order uses its own interface and follows its own pipeline. You can construct pipelines for chained orders that, in turn, spawn another n-tier order. In theory, n-levels of orders can be chained or derived, if applicable within a given scenario.

The creation of the n-tier order is tied to the Create Chained Order transaction which converts the releases on the parent order into a single chained order for a given organization pair. The document type for the n-tier order can be specified during the configuration of the document type of the parent order.

7.6.1 Chained Orders

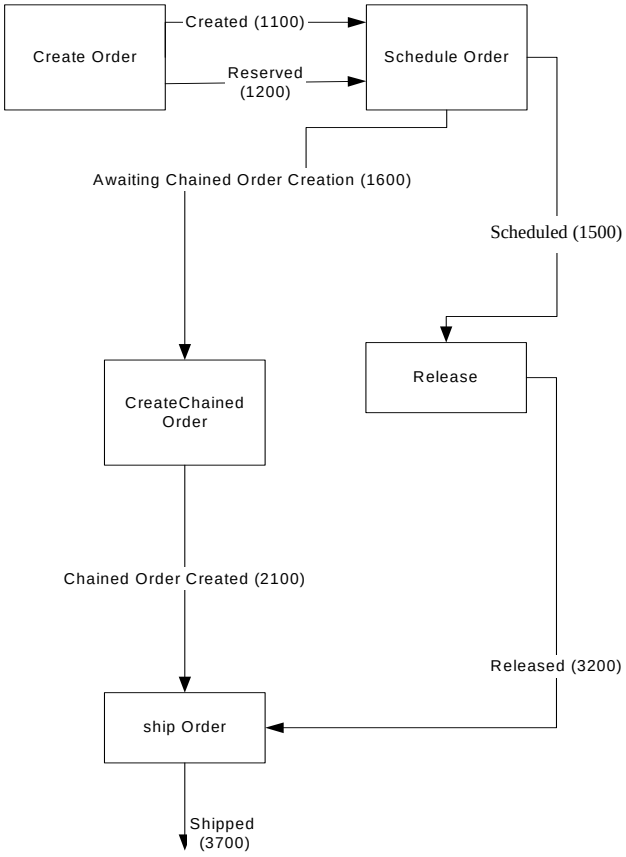
Chained Orders are subordinate orders created as a result of the necessity of the parent order to communicate some portion of the order fulfillment execution to a third party. A tiered order is defined as "Chained" if the subordinate order must finish its fulfillment process before its parent order can be considered fulfilled. For example, an order is placed with an enterprise which, in turn, creates a purchase order with a drop-ship supplier to get the order fulfilled directly to the customer. The orders "chained" to the parent order are responsible for reporting status updates back to the parent order. Any modification made to the parent order after the chained orders are created may be propagated to each chained order.

> **Note:** For chained orders the Seller of the parent order must also be configured as a Buyer.

Essentially, once the decision (either manually or through the scheduling process) has been made to source a particular line from a drop-ship supplier, the line is included in a chained purchase order that, through its independently configured pipeline, can then be fulfilled. The second order that is created through this process is referred to as the chained order.

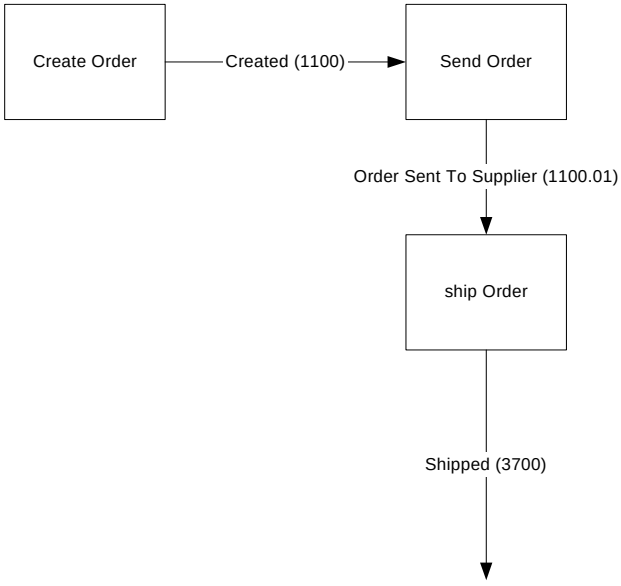
The following figure illustrates the high-level workflow of the original order:

Figure 7–6 Order Workflow Spawning a Chained Order



The Chained Order pipeline can then be constructed as illustrated in the following figure.

Figure 7–7 Chained Order Pipeline



7.6.1.1 Chained Order Creation for Services

Chained orders can also be created for service lines using the IncludeInChainedOrder transaction. The service line of a chained order can be either a provided service or a delivery service. It can also be a standalone line or can be associated with a product. When a service line is chained, neither the associated lines nor the association among the lines is carried over to a chained order. Service lines cannot be chained partially.

A chained service line can be cancelled, included in a work order, or completed. When a chained service line is completed, the parent service line is also marked completed. A complete service line does not mark the product line completed automatically. If a chained service line is completed as part of a work order, the chained line as well as the parent line are marked completed through the deliver order transaction. The service line can be completed either through the work order or through

the chain order. The chained order cannot be created if a service line is included in the work order and vice versa.

> **Note:** The DELIVER_ORDER transaction must be included in both the Parent and the pipeline of the Chained order pipelines if you have work orders. If it is not included, you receive an error stating there is ‘Not Enough Quantity’ when you confirm the work order for chained order.

7.6.2 Derived Orders

Derived Orders are subordinate orders created from the parent order, but once created they maintain no more than a reference to the parent. Their lifecycle is independent from the parent and they provide no status propagation to the parent order, nor are they updated with any change to the parent order. A tiered order is defined as "Derived" if it does not have to finish its fulfillment process before its parent order can be considered fulfilled. For example, a purchase order can be "derived" from a planned order.

Typically, either the lifecycle of the parent order is already over by the time a derived order is created from it (for example, creating a fulfillment order from a planned order) or a completely new order needs to be generated for some parallel processing (for example, creating exchange orders for returns before the return disposition is done). A link is maintained between a derived order and parent order only for tracking progression of the parent order into other various order documents.

The creation of derived orders, requires that two abstract transactions, "Include in Derived Order" and "Create Derived Order" be created. These transactions can be copied during transaction configuration to create appropriate meaningful transactions. Both transactions can have multiple pickup statuses and a single drop status.

Note: An order can have multiple derived orders out of the same quantity but since chained orders are created to complete the fulfillment of the parent order, a unit in the parent order can belong to one and only one chained order.

7.7 Custom Orders

Custom Orders support supplying customized products. Selling and Fulfillment Foundation supports two types of Custom Orders:

- Orders which are *made-to-order*
- Orders which are *made-to-customer*

A *made-to-order* order has items which are produced uniquely for this order. The item is made from unfinished materials, specifically for this order. There is no existing inventory for the item, and there is no expectation that another item exactly like this one might be ordered.

A *made-to-customer* order is created based on the requirements of the Buyer. It assumes that the buyer plans to place multiple orders for this particular item. The made-to-customer order is designed for use by Buyer organizations that make repeated purchases for an item that has been configured to its requirements.

7.7.1 Made-To-Order Orders

A made-to-order (MTO) order is an order for a item that is created or configured for this order. For example, a computer vendor may sell custom computer configurations, where the buyer can specify a choice of hard disk size, processor type, DVD player, and other hardware choices. The computer vendor assembles the computer from existing components, and ships the configuration that the buyer has requested.

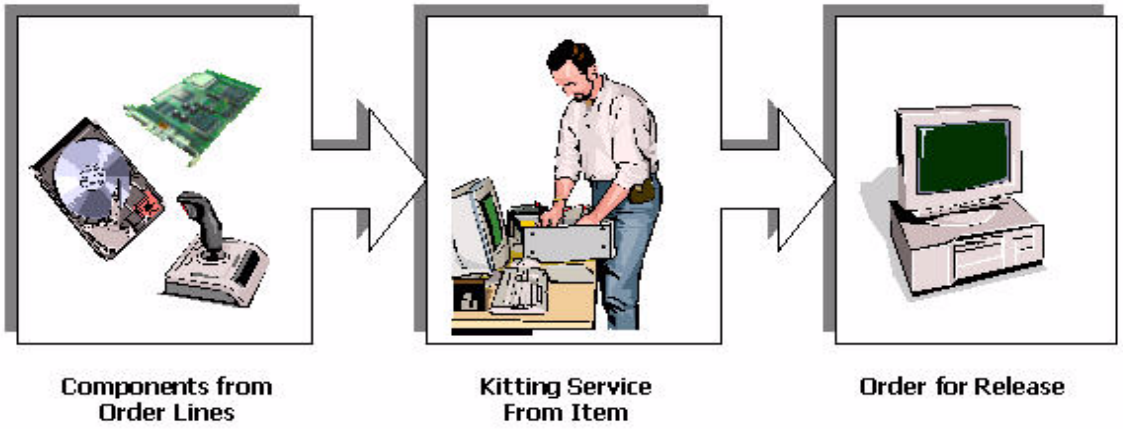
The quantity for an item that is on a made-to-order order may be greater than one. For example, a shirt manufacturer may take an order for two thousand blue shirts which have the corporate logo of the buyer embroidered on them. This is also an order which is made-to-order.

When an order is made-to-order, the item being purchased does not have a definition of its components defined in the catalog.This reflects that the item is being created especially for this order. It may have a Code of "Dynamic Physical Kit" (DPK).

When the order is created, it is assigned a Kit Code of "Dynamic Physical Kit" (DPK), and the components that make up the item are specified in the order lines. The Segment is set to the OrderLine Key, and the Segment Type is set to "MTO", indicating that this is Made To Order.

When scheduling the order, sourcing rules for individual lines are used, and scheduling attempts to located a single ship node. If a single node can not be specified, the item is backordered. For more information about order scheduling, see Chapter 9, "Order Promising and Scheduling".

Figure 7–8 Made-To-Order Order



The creation of a made-to-order order also generates a work order document. A work order document is used to manage a series of actions to fulfill the custom order. For more information about work orders, see Chapter 10, "Value-Added Services".

7.7.2 Made-To-Customer Orders

To support Buyer compliance, an organization may offer Value Added Services (VAS), which provide customization of items to Buyers.

For example, a cell phone manufacturer, *ABC*, may supply a cell phone to two different cellular service providers. The first cellular service provider, *EFG* wants the EFG logo to be applied on each phone. The second cellular service provider, *XYZ* targets a teen market and wants a logo of a popular teen singer applied on the phone. In addition, KLM want special

marketing materials placed in the phone's box. These materials describe special services that appeal to a teen market, such as games, instant messaging, and a music review service.

The cell phone manufacturer, *ABC*, can accommodate these requirements by setting up separate inventory segments for the two Buyers. Unlike the *made-to-order* model, where there is no assumption that an order for a similar object is placed, here the cellular service providers can place multiple orders for their customized cell phone. *ABC* can build up inventory for the cell phones.

To support these Value Added Services, a Buyer Organization is configured to support made-to-customer orders, by setting the Requires_VAS_Compliance flag in the configuration of the Organization. This indicates that the Buyer may have orders for items that are made to their specifications.

When items that can be customized are specified in the catalog, the item is assigned classification of "VAS" (Value Added Service), to indicate that the item is customized to the requirements of the Buyer. In the definition of the item, a compliance services is specified. A compliance services define activities that should be performed to prepare the item to meet the requirements of the buyer. Items which are classified as VAS can not have a Kit.

When an order is created, the Segment is the Buyer Organization, and the Segment Type is "MTC" (made-to-customer) indicating that this a a made-to-customer order.

When a made-to-customer order is submitted, if there is sufficient inventory for the order, a work order is created.

A work order confirmation may result in inventory transformations. Selling and Fulfillment Foundation provides visibility into such inventory transformations using the Allocation Considerations configuration.

Work order allocation in Selling and Fulfillment Foundation is used as follows:

- When a work order is created, demand is placed against the original inventory (the one being consumed), and supply is increased for the new inventory (the one being created). However, the supply being increased is not an onhand supply. It is an indicative supply that would be available in future. And, the demands being increased are not promised demands.

- When a work order is allocated, the demands placed are modified to indicate that the demands are promised. The supplies may also be modified to indicate their increased chance of arrival. These demands and supplies could be used to assess the availability of inventory.
- When a work order is confirmed, the supply for the original inventory is removed and supply for the new inventory is created.

For more information about work order execution, see Chapter 10, "Value-Added Services".

7.8 Order Hold Processing

Selling and Fulfillment Foundation allows for one or more hold types to be applied to an order. A hold type prevents an order from being processed by transactions that are associated with that hold type.

A hold can be in three different statuses:

- `Created`, when the hold has just been applied to an order, and has not yet been examined.
- `Rejected`, when the hold has been reviewed by a supervisor who decided that the order should not be processed.
- `Resolved`, when the hold has been reviewed by a supervisor who decided that the order should be processed.

Order Hold types can be applied in the following ways:

- Manually: a supervisor may feel that an order needs to be placed on hold for fraud check, and applies the hold through the Application Console. It is possible to configure a particular hold type so that only users of a specific group, or set of group, can apply the hold to an order.
- Automatically at draft order creation: every time a draft order is created, it is placed on a specific hold type by default.
- Automatically at draft order confirmation, or order creation: every time a draft order is confirmed, or an order is created without going through the draft status, it is placed on a specific hold type by default.
- Automatically upon resolution of another hold type: if a certain hold type is resolved, it can trigger another hold type automatically. This is

specified in the hold type that is being applied on resolution of the other.

- Automatically when a specific modification type occurs: hold types can be configured to be placed automatically when a certain modification type occurs at the order, order line, release, or release line levels, for instance a fraud check hold whenever a new payment type is added.
- Conditionally: Holds can be applied at either Order or Order Line level that meet a certain condition.
- Automatically when an order requires approval: if an order meets the conditions necessary to require approval, a configured hold type is triggered.

> **Note:** When a draft order is created, the order is not placed on hold, even if the order requires approval. However, when a draft order is confirmed, and the order meets the conditions necessary to require approval, the order is placed on hold.

Independently of how the hold is applied, it is possible to specify a condition that determines whether or not the order should be put on hold. For example, you could want to only place orders with a specific payment status on hold. You can use the condition builder to do this on a hold type. For more information about using the condition builder, see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide*.

By default, all transactions are allowed to process an order or order line that is on hold. Using the Applications Manager, you can specify which transactions are prevented from processing an order that is in a particular hold type. Transactions that can be configured to not process orders on hold are known as hold type enabled transactions, and are defined at the document type level for a given enterprise. Custom transactions that are not derived from an abstract transaction are all hold type enabled.

Two transactions need to be configured to process orders or order lines that are in a hold type, and remove them from that hold: one for holds in `created` status, and one for holds in `rejected` status. Holds can be resolved manually, and it is possible to specify a set of user groups with

the authorization to do so. That too is done at the document type level for a given enterprise in the Applications Manager. Additionally, based on a configuration rule, order approval holds can be resolved automatically when an order is changed.

7.8.1 Order Line Hold Types

You can also apply holds to single lines on an order. Functionally, Order Line holds are similar to Order Level holds. For example, when an order that contains multiple lines is created, you may want to perform a Fraud Check on each line. By enabling the Fraud Check hold to be applied to each line, lines which have passed the check are able to move on through the fulfillment process.

Order Line level holds can be applied in the same manner as Order Level holds, with the following additions:

- Automatically upon the creation of a draft order that contains lines, or when a line is added to a draft order.
- Automatically when a line is added to a regular order.

Hold Effect Level

As with Order Level Holds, you can define the effect that applying the specified hold has on the order line. However, Order Line Level Holds also have the ability to prevent transactions from processing at either order or line level. For example, when the ship node of any line changes, you could automatically apply the SHIP_NODE_HOLD hold. This hold can be configured to prevent the Schedule transaction from occurring for either the specific line, to allow the rest of the order to be processed, or the entire order can be stopped due to the change.

Note: You can prevent the following transactions at the Order level only. They process all lines on an order together.

- Payment Execution
- Payment Collection
- Close Order
- Send Release
- Confirm Draft Order

Setting the Hold Flag in the Task Queue

The hold flag in the Task Queue can be set for Order Line holds based upon the following scenarios.

- If all the lines on an order that are eligible for a Task Queue based transaction have a hold applied which prevents that transaction, the hold flag in the Task Queue is set to 'Y'.
- If an order contains a line with a hold that prevents an Order level Task Queue transaction, the hold flag in the Task Queue is set to 'Y'.

7.9 Multi-Client Orders

When multiple versions of PCAs are installed on one version of Selling and Fulfillment Foundation, an order created in one version of a PCA can be accessible in another version.

This enables customers to roll out a new version of a PCA across one set of stores, while the others remain on the older version until ready to upgrade.

In this scenario, a Customer Service Representative (CSR) can create an order in using the Sterling Call Center and Sterling Store Release 8.2, and another CSR is able to schedule pickup on that order in Release 9.0. When a feature has some access limitation between versions, a message may be provided to the user indicating that this feature is limited or unavailable across versions.

The multi-client order feature is enabled by setting up order tags, which are triggered automatically on certain modification rules. During order creation or modification, these tags are automatically set.

Tag creation on an order occurs based on the following criteria:

- It satisfies the configured Order Header condition
- It satisfies the configured Order Line condition
- Specific modification types have occurred
- They can be set manually

For more information about defining qualified tags, see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide*. For information about configuring order tags, see the *Sterling Distributed Order Management: Configuration Guide*.

7.10 Inventory Reservation During Order Capture

Business requirements often mandate that product inventory is guaranteed for a customer and put on hold when the order is received. This is achieved by reserving inventory for a customer.

Reservation is the process of holding inventory for a customer who shows the intent to buy it later. This ensures availability of the desired items for the customer during order capture. An order can be reserved against an existing inventory reservation or an attempt can be made to make new reservations for the order.

When using an existing inventory reservation, a reservation must be created in the system for the specified item before the order is entered into the system. This reservation is then specified at the time of order creation. The inventory being held is then consumed by the order.

If a reservation does not exist for a specified item, the system attempts to check for inventory and reserves it for the order if it is available.

On similar lines, Capacity allocation can be done for Delivery and Provided Service Lines (similar to inventory reservation for Product items). Refer to section on Capacity Allocation during Order Capture for more details.

Note: Selling and Fulfillment Foundation does not currently support order reservations for time sensitive inventory.

In addition to the reservation process described above, Selling and Fulfillment Foundation provides the following additional capabilities around reserving the order:

- Reservation of Draft Orders
- Reservation on Procurement Node
- Order Line Creation on Reservation Only

Reservation of Draft Orders

A draft order is an order that has had data created for it, but has not been confirmed in the system. These may be order proposals or orders that show an intent to buy on part of the customer. Order execution processes cannot be performed on a draft order until it is confirmed.

Selling and Fulfillment Foundation provide the capability to hold inventory for draft orders through a reservation. If inventory is available in the system, a draft order can be created in 'Draft Order Reserved' status. When created in this status a 'Reserved' demand is created in the system.

Reservation on Procurement Node

Orders can be created with a specified procurement node. When a procurement node is defined for an order line, it must be procured (by means of purchase order or transfer order) from the node. In this scenario, the ship node defined for the order line receives inventory from the procurement node to fulfill the order.

When an order line is associated with a procurement node and requires reservation, inventory must be reserved at the procurement node. In this scenario, 'Reserved' inventory demand is created for the procurement node as well as the ship node. A supply is also created at the ship node depicting the incoming supply at the ship node from the procurement node. When the order is scheduled and a procurement order is created, the extra demand on the procurement node and supply at the ship node is removed.

Order Line Creation on Reservation Only

Occasionally an order or order line is only considered if inventory is available to fulfill the order. Selling and Fulfillment Foundation supports this functionality by ensuring that an order line is created only if a reservation can be made for it at the time of order creation. After order creation, any change to the order line that requires an inventory check is only allowed if inventory is available for the changed attributes. A partial or multiple reservation to the order line is possible. It provides the ability to hold any reservation associated to the line irrespective of the node, the item, or the dates that are on the line. Reservations are allowed against different attributes from the order line. If the node on the line is Node1, reservations are accepted against Node2. The attributes on existing line reservations cannot be modified except for quantity. The quantity can only be reduced or the reservation can be deleted. (If reserving on the fly, we can also increase quantity).

Line level modifications such as quantity additions, node change, and so on do not impact the associated reservations. Only when the line is cancelled completely, its reservations are deleted. Once the line is released, no additional line reservations are allowed. However, prior to the released status, even if the line is scheduled, reservations can be added.

The inventory is reserved by each reservation other than reserving at a line level as each reservation has different attributes at the line itself. However, status on the line is still updated provided that the reservation attributes match the line attributes, for as much reserved quantity that matches the line attributes. In these cases, the order line reservations are rolled up into the line itself. For example, an order is made for a quantity of 10 units of ITEM1 from NODE1, two line reservations are passed, one for 7 units of ITEM1 from NODE1 and the other for 3 units of ITEM1 from NODE2 (perhaps procurement is an option). In this case, the quantity 7 reservation matches the order line attributes, so the reservation is removed from the order line reservation table and quantity 7 on the order line moves to "Reserved" status. The other 3 quantities is in open order status. This causes some reallocation, as the final picture of demand blocked by the order is 7 reserved at NODE1 which is done by the status, 3 reserved at NODE2 which is done by the order line reservation, and 3 open order demand at NODE1 done by the status. Reservations that do not match with the order line are reserved but not rolled up in to the order line. The schedule later looks at these

reservations and determines their validity. However, until schedule runs the demand is blocked for the order line.

7.10.1 Multiple Line Reservations

A partial or multiple reservation for an order line is possible. It provides the ability to hold any reservation associated with the line irrespective of the node, item, or the dates that are on the line. Reservations are allowed against different attributes from the order line. For example, if the node on the line is Node1, reservations against Node2 are possible. Whenever `scheduleOrder` API is run, it attempts to use reservations irrespective of the node by assuming the inventory to be available. All line reservations are removed after scheduling.

Whenever order line reservation attributes match line attributes such as shipping node, item information, and date, the order line reservation is consumed by the line status. In this scenario, the line goes to Reserved status and the order line reservation is not created.

7.10.2 Reservation Parameters

Several parameters can be defined for an Enterprise to handle reservations in various scenarios.

Reservation Required

Reservation Required indicates that if inventory can be reserved for an order, an attempt should be made to reserve it.

Reservation Mandatory

Reservation Mandatory dictates that inventory must be available for reservation for an order line at the time of order creation or order change. If inventory is not available for the order line, it is not created in the system.

Suppress Validation of Reservation on Line Change

Suppress Validation of Reservation on Line Change allows changes to be made to a reserved order without making inventory checks. However, this parameter is used only if reservation is not mandatory.

Use Reservation With Exact Date

Selling and Fulfillment Foundation allows the creation of reservations for a specified date. The inventory is reserved in the system from this date. If this parameter is enabled, only existing reservations whose reservation date is the same as the requested ship date for the order can be used.

However, if this parameter is not enabled, all reservations with a reservation date before the requested date of the order can also be used for reserving the order.

7.11 Capacity Allocation During Order Capture

Selling and Fulfillment Foundation provides the capability to allocate capacity (or promise resource availability) for Provided and Delivery service lines during order capture. This is achieved by checking capacity to perform a Provided or Delivery service during order line creation. If the capacity is available, capacity can be allocated (or promised) during creation of the line. This capacity is then considered unavailable for the other order lines. Modifications to a service line can also be checked for capacity availability. The following key capabilities are provided by Selling and Fulfillment Foundation for handling capacity allocation of delivery and provided service lines:

- Ability to check capacity availability during order line creation and allocate (block) capacity if it is available.
- Ability to allocate capacity after order line creation by providing a valid appointment for the line.
- Ability to change the Promised Appointment for a provided service line any time before line shipment or completion of work order.

Note: Status of a Provided Service line or Delivery line does not indicate whether or not capacity has been allocated for the line. Appointment Status indicates if capacity has been allocated for the service line.

7.11.1 Capacity Allocation Parameters

Selling and Fulfillment Foundation provides the following parameters for handling capacity allocations:

A valid appointment should be present on the order line before any attempt is made to allocate capacity. If the appointment is not available, the attempt is made to allocate capacity after the creation of the line.

During creation, if capacity is not available, the appointment is taken but no capacity is allocated.

Changes to the order line that cause the capacity availability to be reconsidered include:

- Addition to Ordered Quantity or addition of options
- Shipment Address
- Ship node selected for line fulfillment.
- Appointment Date and Time when the service needs to be performed.
- Skill changes made by adding another product line to the existing Delivery Service or Provided Service.

Override Capacity Check

For each service line, a parameter can be specified to indicate whether or not to check capacity during line creation or for any subsequent change. If this parameter states that capacity check is overridden, then capacity is assumed to be available for the service line and no check is made. Capacity is allocated for the appointment dates provided on the line. However, a valid resource pool must be setup which can provide the delivery or provided service desired.

7.11.2 Capacity Reservations

Capacity reservations allow capacity to be blocked before a sales order is created. During order creation, a Capacity Reservation ID can be passed and capacity blocked is consumed by the created order. Selling and Fulfillment Foundation can use a capacity check to ensure that the reservation is being made against capacity that is available.

Capacity reservations are created only if capacity is completely available for the specified slot. For example, if a resource pool defines two slots, 9-12 and 1-5, and the available capacity is four hours in each slot:

Table 7–6

Reservation ID	Slot	Quantity	Result
Res1	9-12	3	Reservation is created.
Res2	1-4	1	Reservation is not created, since there is no slot defined.
Res3	12-6	1	Reservation is not created, since there is no slot defined.
Res4	1-5	7	Reservation is not created, since there is not enough capacity available.

Capacity reservations are supported for both resource pools and resources. For resources, if capacity is blocked against a resource, the actual capacity blocked is based on the calendar of the resource.

Slot based and non-slot based reservations can be created. When a slot based reservation is made against a resource pool that does not maintain capacity at the resource level, a capacity check can be performed to check for available capacity.

When creating a capacity reservation, an expiration date can be passed, such that when the reservation expires it is available for purging. This releases reserved capacity to be allocated or reserved for new orders. Reservation capacity and expiration date can be modified (assuming capacity is available). Reservations can also be deleted manually

7.12 Item Validation

Item validation enables an enterprise to validate if the product items on an order belong to the catalog of the enterprise.

Note: By default, Item validation is Off.

7.12.1 Extended Validations

Extended validations enable an enterprise to perform the following additional item validations on an order:

- Effective date range
- Cannot be sold separately
- Minimum and maximum quantities
- Item status
- Customer entitlements

Item validation must be enabled before the system performs extended validations.

Note: By default, extended validation is enabled.

Note: An order can also be validated by calling the `validateItemForOrder` API.

Effective Date Range

The effective date range of an item in an order is validated when the order is created, modified, or confirmed. If the system date is outside the effective date range of the item, the system throws an error during item validation. However, if supersession is enabled for the item, the validation is ignored.

Cannot be sold separately

The validation ensures that the item in the order is a part of a bundle or a physical kit. Validation occurs when the order is created, modified, or confirmed.

Minimum and Maximum Quantity

The minimum and maximum quantities of an item in an order are validated on order confirmation. If the quantity of an item in an order falls outside the minimum-maximum range for the item, the system throws an error.

Item Status

The validation ensures that the status of the item is orderable at the time of order creation and modification. If the status of an item is not Published, the system throws an error. For more information about item statuses, refer to the *Catalog Management: Configuration Guide*.

Customer Entitlements

Customer entitlements for an item in an order are validated when the order is created, modified, or confirmed. If a customer tries to order an item to which the customer is not entitled, the system throws an error during item validation. For more information about customer entitlements, refer to the *Catalog Management: Concepts Guide*.

Note: If an item is unpublished, expired, or cannot be sold separately, the item will still appear in the product catalog as long as a product association exists where the current date falls between that association's effective date range. However, the item will not be available for ordering.

An item becomes invalid when one of the following criteria is met:

- The current date falls outside of the item's effective date range.
- The product cannot be sold separately (IsSoldSeparately="N").
- The item is not published (status not equal to 3000).

7.13 Pending Changes on Confirmed Orders

Selling and Fulfillment Foundation provides the capability for a user to make changes to a confirmed order, assuming that the order is in a status that allows modifications. To persist the changes, the user must actually save the changes. However, Selling and Fulfillment Foundation also provides the capability for the changes to be rolled back. Any modifications made to a confirmed order remain in a pending state until the user saves or rolls back the changes, or the changes automatically expire after a preconfigured timeframe.

For example, a user can add an item to a confirmed order and model the order. If the user saves the change, the change is permanently applied to the order. However, after modeling the order, the user may decide that the item is too expensive. The confirmed order shows the added item, but the user closes the browser without saving the pending change. After a specified timeframe, which is defined in the Sterling Distributed Order Management Applications Manager, the pending change automatically expires and the order rolls back to its original state.

When a confirmed order contains pending changes, a hold is automatically placed on the order, preventing any further order processing. After the pending changes are saved or rolled back, the hold is automatically removed and order processing can continue. If the pending changes expire, the hold remains on the order until the ProcessOrderHoldType agent is invoked, which automatically removes the hold.

The functionality of pending changes on confirmed orders is similar to draft orders in that users can make changes to orders, model orders, and confirm orders. However, it differs in that holds are not placed on draft orders, and draft orders do not usually expire after users close their browsers. When users log in again, draft orders are usually available for modification and confirmation.

8

Opportunity Management

An opportunity is a container of one or more quotes that are associated with a single sales opportunity. For example, a Field Sales Representative may see an opportunity to sell laptop computers and desktop computers to a specific customer. The Field Sales Representative generates one or more quotes, which are then presented to the customer. For example:

- Quote A: Buy 5 laptop computers and 5 desktop computers for a total cost of $4,000.00.
- Quote B: Buy 4 laptop computers and 6 desktop computers for a total cost of $3,500.00.
- Quote C: Buy 2 laptop computers and 8 desktop computers for a total cost of $3,000.00.

The customer can negotiate the terms of any of these quotes with the Field Sales Representative. However, the customer can accept only one quote, thereby abandoning all other quotes associated with the sales opportunity. If the customer accepts a quote, an order is created from that quote, all associated quotes are abandoned, and the opportunity is won. If the customer does not want to negotiate on any of the quotes or the quotes expire, the quotes are abandoned and the opportunity is lost.

For additional information about creating and managing opportunities, refer to *Selling and Fulfillment Foundation: Setting Up Quotes*.

9

Order Promising and Scheduling

9.1 Order Promising

In the supply chain industry, the products or services requested by means of an order must be checked for availability, promised, scheduled for shipment or delivery, and then released (shipped or delivered).

As a part of its Order Promising functionality, Selling and Fulfillment Foundation provides configurable rules that can be used for controlling the selection of nodes and shipping dates for both products and services. To help you understand the capabilities of these Order Promising features, this chapter describes the Selling and Fulfillment Foundation concepts associated with promising functions and provides information about the required setup to achieve your business objectives.

Order promising functionality is provided for products being shipped as well as service requests (delivery and provided services). For more detailed information about these specific promising features, see Section 9.2, "Promising for Products Being Shipped" or Section 9.3, "Promising Service Requests".

To understand how all of this is accomplished in Selling and Fulfillment Foundation, you should first be familiar with the following common terms associated with order promising:

Node

A node is a physical location to or from where a product is shipped, returned, or delivered.

When a node is specified on an order line, it indicates the intent to fulfill the order line from that node. If the node is a firm predefined node, order promising functionality ensures that the order line is fulfilled from only this node.

However, certain business requirements may require inventory to be reserved for an order as soon as it is created. This inventory reservation is achieved by providing the ship node on the order line where inventory is reserved. The node specified for reservation may not be the most optimal node to fulfill the order. In this scenario, the node can be marked as a non-firm node. When promising the order, Selling and Fulfillment Foundation scheduling functionality tries to find alternate nodes where inventory may be available based upon the sourcing setup.

By default, Selling and Fulfillment Foundation treats each node supplied on the order line as a firm node unless otherwise stated.

Distribution Group

A distribution group (referred to as distribution rule in previous releases) is a set of nodes or organizations defined for distributing products or services.

Availability Inquiry

Most systems provide inventory availability inquiry and service availability functions separately. Typically, you have to inquire within the inventory module to find product availability. A separate capacity module, if available for services, can be inquired for capacity availability. If more complex inquiry functions are required that had to consider ship-to locations or multiple product availability with some constraints, such as must ship together, you have to create a quote on the system before you can query the availability.

Selling and Fulfillment Foundation, however, has taken a different approach and it provides an API to inquire about the availability of multiple products AND services in a single inquiry function without having to create a persisted document on the system. This API considers all the inter-dependencies between the multiple lines when suggesting your possible sourcing options. The same inquiry function is also available if you have created an order in the system.

Available to Promise Rules

Available to Promise (ATP) rules help to set up a monitoring system for tracking inventory item availability and raise specific actions when the inventory falls below a specified minimum level. The availability of an item can be tracked on the current day, subsequent days within the ATP time frame, and subsequent days outside the ATP time frame. This

enables you to more accurately order supplies to meet current and future demand.

Scheduling

Scheduling is the process of:

- Determining the shipping node or supplier for product fulfillment or the service provider for service fulfillment. This logical process step is referred to as "Sourcing".
- Determining the dates when the product or service is shipped or delivered.
- Reserving the inventory at the shipping location for the shipment date. However, if inventory is already reserved for a product line scheduling does not attempt to recheck inventory.

Note that the scheduling function requires an order document.

Scheduling Rule

Scheduling rules control common scheduling parameters such as:

- When an order should be scheduled
- How many days ahead should product availability be checked (if you are promising against future inventory)
- What type of optimization should be used for scheduling

  For example, you can optimize based on date so that Selling and Fulfillment Foundation chooses a shipping node that can deliver on the earliest date. You could also optimize based on the number of shipments; Selling and Fulfillment Foundation minimizes the total number of shipments required even though it may end up getting delivered at a later date.
- Ship complete order or ship complete line parameters

Sourcing Rules

Sourcing rules control which node, external organization, or group of nodes should be considered for sourcing a product based on the product, item classifications, ship-to region, and other parameters.

Notification

Notification is the process of notifying the shipping node, vendor, or service provider when fulfilling the order request. Selling and Fulfillment Foundation creates several different documents for this notification process.

Delivering Node

A delivering node is the location where a product is being delivered using a last mile service.

9.2 Promising for Products Being Shipped

As part of its promising functions, Selling and Fulfillment Foundation provides the following capabilities:

- Inquiries about product availability. The product availability inquiry function takes into consideration all the business rules associated with the promising function and provides one or more options for sourcing and possible shipment dates. Using the user interface, you can inquire about product availability for a specific order. Using the promising API (`findInventory ()`), which does not require a pre-existing order, you can inquire about product availability for a set of items.
- Scheduling an order or order line. Selling and Fulfillment Foundation provides APIs and transactions to schedule an order for shipping at an appropriate time. The scheduling function determines the node and expected shipment date and schedules a shipment against the same. The scheduling function also reserves the inventory at the shipping node for the expected shipment date.
- Creation of notifications and chained documents to notify the nodes or suppliers at the appropriate time.

9.2.1 Finding Product Availability

This section discusses concepts associated with finding product availability.

9.2.1.1 Available to Promise Rules

Available to Promise (ATP) rules enable you to determine the availability of an item for current and future demand. This determination makes the most efficient use of inventory so that items are not set aside for future orders when they could be used to fulfill more immediate demands. The availability of an item is based on current and future supply, lead time, and ATP configuration. The lead time is the amount of time it takes your supplier (distribution center or drop-ship supplier) to purchase an item for shipping. The processing time covers the time it takes for an item to be received by a supplier and made ready for shipment (inbound processing) as well as shipping it from the warehouse (outbound processing).

ATP rules enable you to effectively manage orders for items. Parameters can be set for the amount of time an item is available for current and future orders. With a First Expiration First Out (FEFO) inventory management system, perishable items can be sold and shipped well before their expiration dates, ensuring first expired inventory is consumed first.

9.2.1.2 Inventory Availability Safety Factor

The inventory availability safety factor defines a fixed quantity or percentage that is excluded from inventory availability for various purposes. Application of the inventory availability safety factor is controlled at the scheduling rule, supply type, and node type.

Promising Onhand Inventory

When dealing with store availability in a scenario where availability is shared between online consumers and in-store customers, the actual availability of a product at any given time may not be accurate, due to misplaced items, items in shopping carts, or items on reserve. This may lead to over-promising to online consumers.

Using the inventory availability safety factor, stores can define the quantity or percentage of inventory, by item or inventory supply type, to be excluded from the inventory availability that is visible to online consumers.

Promising Future Inventory

Future inventory is supply that is expected to arrive at a supplier, but is not yet on hand. Future inventory that is generated by a purchase order

can be used by an enterprise to promise against new and existing orders. However, in some cases, not all future inventory can be considered to be completely reliable. For example, if an enterprise has just placed a purchase order to one of its suppliers, it's possible that the order may become backordered. Therefore, the enterprise may only want to consider 40% of the future supply created by the purchase order for order promising. If the supply has been shipped and is in transit, the enterprise can be more confident that the supply is received in a timely manner. Therefore, the enterprise can safely promise up to 95% of the in transit supply.

Selling and Fulfillment Foundation provides for this flexibility. Each enterprise can define the percentage of each supply type that can be used to promise demand.

Unplanned Inventory

The unplanned inventory is the inventory that is assumed to be available based on the lead time of an item. For example, if there is an order inquiry for 10 quantities of item1 with the lead time of 10 days, and there are only 7 quantities of item1 in onhand inventory. When unplanned inventory is considered, the order can be scheduled against the following inventory:

Onhand Inventory = Available on current date for 7 quantities

Unplanned Inventory = Available on current date + 10 days (lead time) for 3 quantities

9.2.1.3 Minimum Ship-By Date

When dealing with time-sensitive items, customers may have different preferences on the life span of their products after they are shipped. For example, a regular retailer may request an item with a remaining life span of 240 days, whereas a discount retailer may request the same item with a life span of 120 days, for less cost.

You can specify a minimum ship-by date on an order line during order creation. The order creation process takes into consideration the preferred remaining life span of a time-sensitive item beyond its ship date. The minimum ship-by date is the preferred ship-by date in addition to the requested remaining life of the product.

During the promising and scheduling processes for a time-sensitive item, Selling and Fulfillment Foundation ensures that the available quantity is

promised only when the minimum ship-by date on the order line falls before or on the same day as the ship-by date of the inventory supply (expiration date).

For example, the following onhand inventory is available:

Supply	Product	Expiration Date	Quantity
Supply1	Item1	7/15/2005	5
Supply2	Item1	7/30/2005	10

An order for the following comes in:

Product	Quantity	Demand Ship Date	Minimum Ship-By Date
Item1	5	7/1/2005	7/21/2005

This implies that to fulfill this order, the quantity of **Item1** promised must have a remaining life span of 20 days at the time of shipment.

Since **Supply1** would expire 14 days after shipment, this inventory would not be considered for promising. In contrast, **Supply2** would expire 29 days after shipment, and thus meets the specified criteria.

9.2.1.4 Sourcing

Sourcing is the process of determining from which node or supplier a product should be shipped.

In the most simplistic scenarios when you have just a single distribution center, you already know the shipping node. In such cases you can pre-specify the shipping node in the inquiry function or specify it on the order line. If this node is a firm pre-defined node, an order line is only sourced from that ship node. You need not do any sourcing configuration in this case.

However, if the node is not a firm pre-defined node, the remaining sourcing setup is used. If the node specified on the line is not a firm node or in more complex situations, you may have multiple nodes and suppliers from which you source the products.

The requirements for selection of the correct shipping location may be based on:

- What product is being shipped.

- Ship to location - typically you would want to ship out of the closest shipping location but you still need to ensure that geopolitical requirements are met. For example, if shipping to El Paso, Texas you may want to ship the product from a distribution center located in the United States even though a Mexican distribution center is closer to El Paso.
- Product availability at different locations. A warehouse may or may not have inventory for a given item. However, a manufacturer may want to always assume that any demand can be fulfilled using existing raw material at the node. To achieve this, Selling and Fulfillment Foundation can be configured to never perform inventory checks on schedule and release for certain items, at any node. You can enable this functionality at the item level.
- Total number of shipments that are required to complete the request. This, in some ways, reflects the transportation and handling costs associated with the entire shipment. If an order is placed for 3 items and one of your nodes carries all three and another node carries only 2 them, you may want to ship from the node carrying all 3 items in just one shipment compared to two shipments in the other case.
- Prioritization of nodes. You may want to first ship out of your own locations and then use drop-ship suppliers when no product is available in your own nodes.
- Customer specified constraints such as ship together dependencies and fill quantities being met from the selected node.
- Ability to perform services for a work order. This could include kitting and dekitting services, and ability to supply services for buyer compliance.
- Ability to perform gift wrapping services. A ship node may or may not offer gift wrapping as a fulfillment option for an item. You can enable this functionality at the node attribute level.
- A customizable order sourcing classification. For example, an enterprise may want to use a customer attribute as a sourcing parameter. The order sourcing classification enables this flexibility.

The following sections discuss the configuration required to achieve these requirements. When making sourcing and date decisions, Selling and Fulfillment Foundation uses the configuration setup for the:

- Enterprise on the order. Exceptions to this rule are noted in individual cases.
- Primary Enterprise of the Organization code passed in on-the-fly inquiry APIs.

9.2.1.4.1 Sourcing Configuration

When making sourcing decisions for your products, think about how you would configure the following parameters. This helps you to define your preliminary product sourcing configuration.

1. Are sourcing rules defined?

   Sourcing rules are defined by the Enterprise. You can use this parameter to specify if the Enterprise defines sourcing rules at all. If you ship only from the distribution centers owned by the Enterprise and the Enterprise has only a few distribution centers, you may not need to do any complex sourcing setups. If you set this parameter as "No", Selling and Fulfillment Foundation sources the product from any of the nodes owned by the Enterprise. It uses the optimization logic to pick the appropriate node based on distance and other parameters.

   Note: This functionality has changed from previous releases (5.0 and earlier). In earlier releases, Selling and Fulfillment Foundation was looking for the setup as defined by the Seller organization on the order. There is no backward compatibility provided for this and you need to reconfigure your sourcing rules to achieve the desired functionality.

2. Is inventory information available to the system?

   This parameter is setup for all organizations that can own inventory. This parameter specifies whether or not an organization makes inventory information available to Selling and Fulfillment Foundation. Information can be made available either through the Sterling Global Inventory Visibility application or through a real-time interface.

If you can make inventory information available to the system, you should set this parameter to 'Yes'. This typically results in better sourcing decisions. If such information is not available, you must set this to 'No'.

Also note that apart from this parameter, Selling and Fulfillment Foundation uses inventory availability as one of the constraints only if the:

- Seller organization on the order publishes inventory information (this parameter is set to 'Yes')
- Document type of the order is set to 'read inventory for scheduling'
- Document type of the order is set to 'update inventory for seller organization'

If any of the above is not true, Selling and Fulfillment Foundation assumes that an inventory check does *not* have to be performed at all and the sourcing process does not consider inventory as one of the determining factors. However, other factors such as distance and node priorities are still considered for ship node selection.

3. Is inventory kept externally?

   This parameter is relevant only if the above parameter "Is Inventory information available to the system" is set to "Yes". This parameter specifies if inventory information for the organization is maintained within the Sterling Global Inventory Visibility application or is made available through a real-time interface.

   A single distribution group can specify:

   - Nodes for which inventory is kept within the Sterling Global Inventory Visibility application.
   - Organizations keeping inventory externally

   When checking for inventory, Selling and Fulfillment Foundation uses this parameter to determine the inventory store and if kept externally, makes a call out to get the product availability.

4. Default distribution node group?

   This parameter defines the default set of nodes or suppliers to consider for sourcing when Selling and Fulfillment Foundation does

not find a matching sourcing rule. Sourcing rule setup is discussed in Section 9.2.1.4.3, "Sourcing Rules". Note that this parameter is only relevant for the Enterprise setting up the sourcing rules.

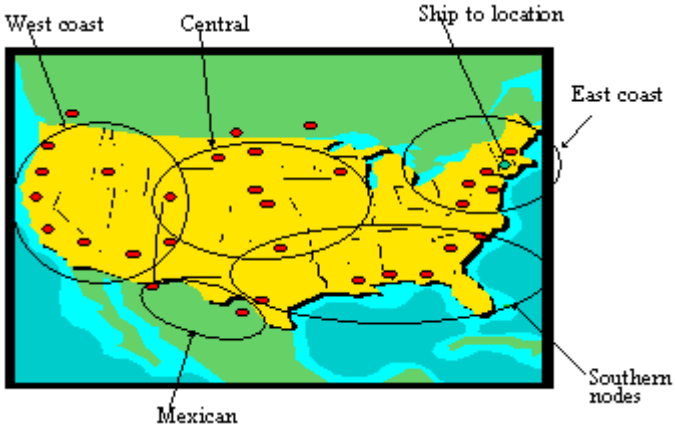
9.2.1.4.2 Distribution Groups

A distribution group is a defined a set of nodes or external organizations. You can then associate the defined distribution group with a sourcing rule. Selling and Fulfillment Foundation considers all the nodes or organizations that are part of this group and optimizes on various factors for making the node selection. A priority number can also be specified for each node or organization. This priority number is used for node selection based on the optimization mode being used for scheduling. For more information about scheduling based on optimization, see Section 9.2.1.4.5, "Scheduling Rules".

For example, in Figure 9–1, "Distribution Group Nodes", multiple distribution node groups such as "East Coast", "Central", and so forth, are depicted. Notice that a single node can be part of multiple node groups at the same time.

Also, note that it is not always necessary to create distribution groups because of proximity of shipping locations. You may create groups to differentiate between your own nodes and external suppliers and then use the sourcing rule's sequencing feature to first try and source from your own nodes. If not enough inventory is found, Selling and Fulfillment Foundation looks into the supplier nodes.

Figure 9–1 Distribution Group Nodes



9.2.1.4.3 Sourcing Rules

If you set the "Sourcing rules defined" parameter to "No", the setup of sourcing rules is irrelevant for you and can be skipped completely. Selling and Fulfillment Foundation automatically considers all the nodes owned by the Enterprise as potential candidates for sourcing the product and selects the optimal node based on the rest of your configuration.

This section discusses sourcing rule setup and how Selling and Fulfillment Foundation uses this configuration.

A sourcing rule can be created by specifying one or more of the following key parameters:

- Item Classifications or Item ID
- Geographical region of the ship-to location or ship-to node
- Fulfillment type
- Seller organization
- Sourcing criteria

You have the flexibility to leave any of the above parameters (except fulfillment type) as void in the sourcing rule and that implies that the sourcing rule is applicable to all values of that parameter.

For each sourcing rule, you can then specify a sequence of node or distribution group to be used for sourcing the product.

Order Sourcing Classification

Selling and Fulfillment Foundation provides the flexibility to configure a custom parameter that is high in the sourcing priority ladder. For example, an enterprise may be supplying products to a number of retail stores, and one of those retail stores does not want their products to come from a warehouse that is in a foreign country. This could be due to the complications that come with crossing borders (duty, delays with merchandise check). It is possible, by specifying a customer name or customer attribute such as the order sourcing classification, to take this restriction into account in the sourcing logic.

Item Classifications or Item ID

Selling and Fulfillment Foundation provides the flexibility to configure sourcing rules based on multiple item classifications to avoid extensive setups required at each item level. At the same time, it also provides the flexibility to specify this setup at an item level for special situations.

One approach is Selling and Fulfillment Foundation could take is to use only a pre-defined "classification" such as product line for this setup. In different situations, you may want to create your own classifications and use them for setting up sourcing rules. For example, in a process industry dealing with steel, the classification could be the "grade of steel, whereas for a retailer it could be "product line".

To provide this flexibility, Selling and Fulfillment Foundation enables you to pick your own item attributes that are used for setting up the sourcing rules. Your catalog organization (the Enterprise could be maintaining its own catalogs) can select the classifications that should be used for purposes of "Product sourcing". For example, your organization can choose to set up sourcing based on a product line classification. Another enterprise in the same installation could choose some other custom classification for sourcing.

Geographical Region

Selling and Fulfillment Foundation provides the flexibility to create sourcing rules based on the region to which the product is being shipped. An organization can choose the Region Schema to be used for determining the shipping region. Every ship-to address is translated to a

shipping region hierarchy in the Region Schema based on its postal code. This shipping region is used to look up the correct sourcing rule. Example 9–1, "Sourcing Rules" and Example 9–2, "Sourcing Rule Region Hierarchy" below illustrate the usage of regions in sourcing rule determination. See Section 9.3.1, "Regions and Region Schema" for a complete understanding of this function.

Procure for Shipment Setup

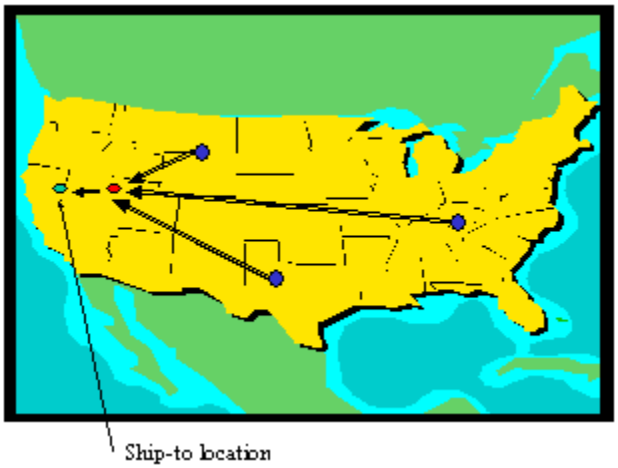
When evaluating product availability, the system considers each sourcing rule detail in the selected shipping sourcing rule. If the product is not available at the node or distribution rule with lowest sequence, the procurement sourcing rule is considered, and evaluated in the order of the sequence specified. If all procurement details have been evaluated but inventory cannot be found, the next detail in the shipping sourcing rule is considered.

Note: Procurement sourcing rules are not supported for logical kit items.

In some situations you may want to ship a product from a particular location even when there is no product available at that location. For such situations, Selling and Fulfillment Foundation can automatically create a transfer or purchase order for the product when it is not available at the shipping location of choice. To do so, the "Procure for shipment" flag must be set to "Yes".

An example of a situation where you would want to do this is when you want to ship a complete order out of a single node close to the final ship-to location but the desired ship node does not stock all of the items or is just a cross-dock location. You set this parameter to "Yes". Selling and Fulfillment Foundation then sources the order to the desired ship node location and then creates a transfer order or a procurement order from other locations to this location to complete that particular order. These "chained procurement orders" act as transfer orders or could be purchase orders that are tied to the customer shipping order. Figure 9–2, "Procurement Distribution Setup" illustrates a typical setup for shipment procurement.

Figure 9–2 Procurement Distribution Setup



This can be very effectively used for implementing "merge in transit" functionality. The merge location can act as the shipping node for the order. Selling and Fulfillment Foundation creates transfer movements to the merge location based on this setup. Selection of the merge location itself can be driven out of the geographical dimension of the sourcing rules.

Apart from this being set to "Yes", the node in question should also be configured to set the "Procure for shipment allowed" flag to "Yes".

To effectively achieve the creation of transfer or procurement orders, the following must also be set up:

- "Procure for ship" sourcing rules

  These are sourcing rules set up by the procuring node. These rules are similar to the other sourcing rules discussed here. You can define multiple sourcing rules with corresponding sequences, similar to shipping sourcing rules. Each sourcing detail contains either a single node or a distribution rule (optimized based on node availability of the selling node), which enables you to consider multiple nodes for procurement. Selling and Fulfillment Foundation looks into the inventory of the node or nodes from where you want to procure items, and if available, creates a procurement order.

- The "Requires chained orders" flag must be set to "Yes"
- Transfer schedule

  Though not mandatory, this schedule can be created to specify a transfer schedule between any two nodes on a "day-of-week" basis. Transit time can also be specified. This setup is used for calculating expected dates and is discussed in Section 9.2.2, "Calculating Expected Dates".

> **Note:** When calling the inventory availability API, Selling and Fulfillment Foundation takes all of this information into account and suggests dates considering that a procurement order would be created. However, Selling and Fulfillment Foundation does not actually create the procurement order until you use the scheduling function to schedule an order.

Fulfillment Type

An Enterprise can define the list of valid values for fulfillment type and specify it on the order line. Selling and Fulfillment Foundation uses the sourcing rule associated with the fulfillment type to determine the correct sourcing rule. A blank value can be specified for the fulfillment type on the order line, but it cannot be blank in the sourcing rule. If it is blank on the order line, the enterprise on the default fulfillment type of the order is used to select a sourcing rule. Selling and Fulfillment Foundation does an exact match between the value of the parameter in the sourcing rule and the fulfillment type on the order line. Unlike other sourcing rule parameters, a blank value in this parameter does **not** signify that it can be used for all fulfillment types.

This control can be used to accommodate custom requirements where you need to use different sourcing locations based on parameters Selling and Fulfillment Foundation either does not understand or does not provide control on in this release; such as customer, quantity of an order line, or order type. You can translate your requirements into different fulfillment types and thus get different sourcing rules.

Seller Organization

Selling and Fulfillment Foundation provides the flexibility to source a product differently based on the Seller organization involved in the

transaction. You may model your stores as different Seller organizations and may want different sourcing based on the store from where the product was sold.

9.2.1.4.4 Sourcing Rule Determination

You can easily notice that you can create conflicting sets of sourcing rules by leaving some of the parameters void. Selling and Fulfillment Foundation resolves such conflicts by using a priority order for the key parameters. The priority order Selling and Fulfillment Foundation uses is:

1. Seller Organization
2. Order Sourcing Classification
3. Item ID
4. Primary Item Classification
5. Secondary Item Classification
6. Tertiary Item Classification
7. Geographical Region of the ship-to location

Fulfillment type is matched exactly with the order line parameter or inquiry parameter.

If you specify just "Item ID" for one rule and "Primary Item Classification" for another rule and both rules match the values for the product being sourced, Selling and Fulfillment Foundation chooses the rule specified for the Item ID over the one specified for Primary Item Classification. It also gives preference to the rule where a value is specified for the parameter over the rule where the value is left blank. Example 9–1, "Sourcing Rules" further explains this concept.

Example 9–1 Sourcing Rules

This example describes a basic sourcing scenario. Your sourcing rule parameters are defined as described in Table 9–1, "Sourcing Rule Parameters for Preference Example".

Table 9–1 Sourcing Rule Parameters for Preference Example

Rule#	Item ID	Primary Item Classification	Geographical Region
1	ITEM1	-	-
2	-	TV	-
3	ITEM1		California
4	-	-	California
5	ITEM1		San Francisco, CA

If you are sourcing an order line with its Item ID, Item Classification and Shipping Location values of ITEM1, TV, and New York, Selling and Fulfillment Foundation uses rule #1. Even though rule #2 also meets the criteria, rule #1 is more specific over the other rule. The Item ID parameter is given higher preference over the Primary Item Classification parameter.

If you are sourcing an order line with its Item ID, Primary Item Classification, respectively, and Shipping Location values of ITEM1, TV, and Los Angeles-CA, Selling and Fulfillment Foundation uses rule #3. Even though rules #1, 2, and 4 also meet the order line criteria, rule #3 is given higher priority over the other rules because each of these rules have some parameters as blank (void) and rule #3 does not. The Item ID parameter is given higher preference over the Geographical Region parameter.

If you are sourcing an order line with its Item ID, Primary Item Classification, and Shipping Location values of ITEM1, TV, and San Francisco-California, respectively, Selling and Fulfillment Foundation uses rule #5. Even though rules #1, 2, 3, and 4 also meet the order line criteria, rule #5 is given higher priority over the other rules because each of these rules have some parameters as blank (void) compared to rule #5. Rule #5 is given higher preference than rule #3 because the region specified in rule #5 is more specific compared to rule #3's region.

Example 9–2, "Sourcing Rule Region Hierarchy" further illustrates the hierarchy for determining the sourcing rules with defined regions.

Example 9–2 Sourcing Rule Region Hierarchy

When selecting a sourcing rule, the rule matching the lowest level region is given higher preference over any rule that has a higher-level region specified.

A hierarchy of region is defined as:

Assume that the sourcing rules are defined as follows:

- Use "Node1" for shipping to any address in New York
- Use "Node2" for shipping to any address in Downtown Manhattan.

The Selling and Fulfillment Foundation node selection for the shipping address chooses Node2 when shipping to Downtown Manhattan and Node1 for any address in Manhattan other than Downtown Manhattan.

Using Sequence of Sourcing Templates

For each sourcing rule, you can specify a sequence of sourcing templates to be used for sourcing the order line.

Selling and Fulfillment Foundation tries to source the product from the highest sequence (lowest number) sourcing template. When multiple choices are available to Selling and Fulfillment Foundation, node selection is optimized based on settings in the scheduling rule associated with the order. Optimization settings are discussed in detail in Section 9.2.1.4.5, "Scheduling Rules".

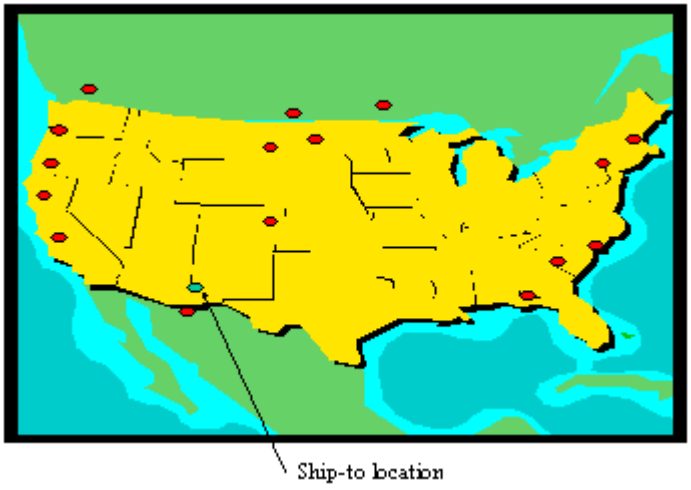
If there is no available product in the sourcing template specified in this sequence, Selling and Fulfillment Foundation tries to source from the next sequenced sourcing template.

When would you want to use sequencing?

In certain situations you may want to have more control of the optimization logic used by Selling and Fulfillment Foundation for sourcing.

For example, you need to ship to a location in El Paso, Texas. You have multiple warehouses in North America. You want to first look into United States warehouses and only if there is no inventory available in the United States warehouses would you want to look in the Mexican warehouses. If you specify all North American locations into the same sequence, Selling and Fulfillment Foundation optimizes on the distance between the ship-from and ship-to locations (this can also be controlled to optimize on other parameters) and that may result in shipment from a Mexican warehouse. Being able to control the sequence of distribution groups to consider enables you to handle such geopolitical situations. Figure 9–3, "Sequencing Sourcing Templates" illustrates this concept.

Figure 9–3 Sequencing Sourcing Templates

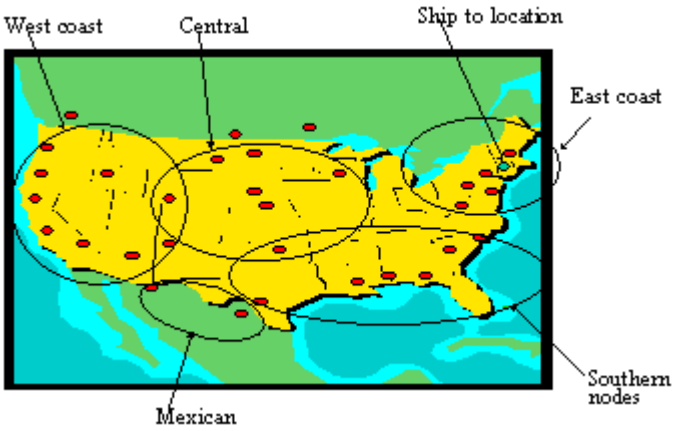


Another reason to provide specific sequences is for system performance reasons.

Suppose a shipment needs to be sent to a New York address. You have 30 to 40 distribution centers spread across the country and you want to make the shipment from the closest distribution center that has the product. You could group your shipping nodes into 4 to 5 distribution node groups based on geographical regions such as east coast nodes, west coast nodes, central US, and southern US.

For this particular situation you would set up the sequence so that Selling and Fulfillment Foundation first tries to source from the East Coast nodes and then the subsequent regions. This is illustrated in Figure 9–4, "Sequencing Sourcing Templates - Example 2". Notice that the same node can be included as part of multiple groups.

Figure 9–4 Sequencing Sourcing Templates - Example 2



One obvious question in the above setup is why shouldn't you just specify a single sequence and associate a distribution node group that includes all United States locations? Since Selling and Fulfillment Foundation already optimizes to select the closest location, this requires less setup with the same results.

The primary reason why you would want to set this up as multiple sequences is to prevent Selling and Fulfillment Foundation from performing redundant inventory searches and thus obtain better response times from the promising functions. Since Selling and Fulfillment Foundation optimizes the selection between all the specified

nodes, it needs to read inventory in all the specified nodes before it makes a final selection. If all the United States locations were to be specified, Selling and Fulfillment Foundation reads the inventory of every location before making the final selection. If you can create a smaller subset of nodes to look into, it in all likelihood save you system resources and provide better response times. The choice between the lesser setup and better performance is mainly based on how many locations you are searching. It is reasonable to put 5 to 10 locations in a single sequence. Any more locations in a single sequence could result in some performance degradation.

Priority of Nodes When Using Multiple Sequences

Selling and Fulfillment Foundation looks at each of the sequences set up in the sourcing rule to source the product until its finds product availability. The priority of an individual node at any given time is calculated as the highest priority of the node amongst all the groups in the sequence. If a node is explicitly mentioned as one of the sequences, priority is treated as 0 (highest).

For each sourcing rule, you can also specify a sequence which enhances scheduling that considers future inventory based on sourcing rule configuration. If preferred substitute is configured for an item, then that needs to be shipped before the onhand inventory. For example, consider:

Sequence1: Source from Regional DC

Consider future inventory which is coming in the next 5 days

Substitution Allowed.

Sequence2: Source from Regional DC

Consider future inventory which is coming in next 30 days

Substitution Allowed.

The Onhand inventory is 5 and the Preferred substitute is 5 which would be coming in 2 days and 10 in 20 days.

If Customer1 orders 5 items, then the preferred substitute is shipped to the customer as it is coming within 5 days.

When Customer2 orders 5 more items, then the onhand inventory is shipped as the preferred substitute now has 0.

Now, when Customer3 orders 5 items, he gets them within 30 days as the preferred substitute and onhand inventory now has 0.

Therefore, these configurations prioritize selection of additional nodes and looking at future inventory.

9.2.1.4.5 Scheduling Rules

This section discusses the parameters related to the sourcing of products only. There are other parameters defined as part of scheduling rules that are used for date determination purposes which are discussed in Section 9.2.2, "Calculating Expected Dates".

Scheduling rules are set up at an Enterprise level. Selling and Fulfillment Foundation uses the rule defined by the Enterprise of the order transaction. When using an on-the-fly inquiry, Selling and Fulfillment Foundation uses the rule defined by the primary Enterprise of the organization code making the request.

Use geography?

This parameter can be used to turn node prioritization on or off based on the distance between the node and the ship-to location. If "Use geography" is set to "Yes" and you are optimizing node selection based on "Priority", Selling and Fulfillment Foundation calculates the node priority as:

Weight factor for distance * distance in miles as calculated based on longitude and latitude + Weight factor for priority * priority setup in the distribution node group. If using multiple sequences, see "Priority of Nodes When Using Multiple Sequences".

This combined priority is used to select the node that has the lowest priority number. The weight factors are also set up as part of the scheduling rules.

If you want to give first preference to the node priority setup, you would want to set up the weight factor of "Priority" as 100,000 irrespective of the distance that is given the first preference. When the priorities of two nodes are the same, the distance is the tiebreaker. If you want to give distance more importance, you set up the weight factor for priority as 0 or another low number. You can work out the weight factors and priority numbers you want to use based on the fact that the distance (in miles) calculation is done internally.

Optimization Type

When multiple nodes and dates are available for sourcing, this parameter can be used to make sourcing selections based on:

- Priority of nodes
- Distance of nodes from the ship-to location
- Date when the delivery can be made
- Number of resultant shipments
- Cost-based scheduling

The optimization type that is required can be set as part of the scheduling rules. The following optimization types can be set:

- Priority - If this is set, node selection is based on the priority setup in the distribution group and the distance from the node to the Ship-To location. Priority is calculated using Weight of node priority * Node priority + Weight of distance * Distance from the node to the ship-to location. Distance calculation is done based on the longitude and latitude of the two locations, and is only a "straight-line" distance. Distance calculation has an approximately 20% error margin from the actual road distance in most scenarios.
- Distance of nodes from the Ship-To location - When optimizing a scheduling solution, the schedule can take into consideration the node priority cost. The priority of the node is converted into schedule cost using the priority cost factor. The product of the node priority and the priority cost factor results in additional cost for using that particular node. For example:

  Distribution Group Passed on Line DG1: N1 – Priority 1, N2 - Priority 10.

  Final leg shipping cost from N1 is $10, from N2 it is $2 (assuming no other costs are being used).

  The optimum solution is N2.

  After the node priority cost factor is enabled,

  Priority Cost Factor - $1

  Option1 - N1 - 1*$1+$10=$11

  Option2 - N2 - 10*$1+$2=$12

The optimum solution is N1 as it has the lesser cost and a low priority of node.

- Date - Selling and Fulfillment Foundation selects the node that can make the earliest delivery of the product. Delivery date is calculated based on the transit time calculated between the shipping and ship-to locations. For more information about transit time calculation, see Section 9.2.2.2.2, "Determining Transit Time".
- Number of shipments - Shipment date and node selection is done in a way that it reduces the total number of shipments finally made. This is also accomplished by consolidating the shipments against future inventory. As a result, the shipment might get delayed beyond what is informed to the customer. To address this issue, you can configure maximum number of days (delay window) by which a shipment can be delayed.

Figure 9–5 Date Range Optimization Logic

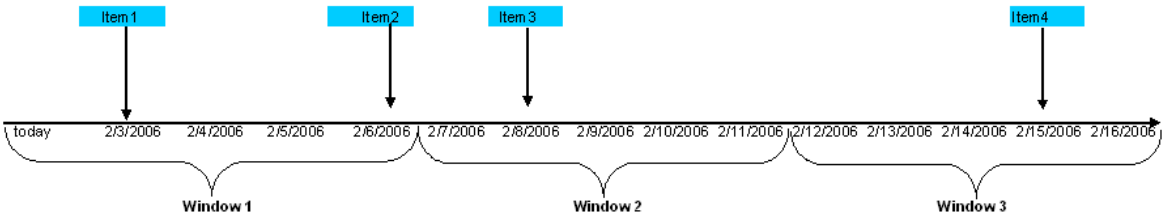


Figure 9–5 illustrates how the windows refer to the optimization date range, which is configured as 5 days. The items are available on various dates. Item1 is available on 2/3/06 and continues to be available until 2/15/06. Similarly, the Item2 is available on 2/6/06 and Item3 on 2/8/06 until 2/15/06. Item 4 is available only on 2/15/06. If the customer places an order for all the items, and has agreed to wait for the shipment, the items are consolidated in one shipment and is shipped on 2/15/06 which reduces the cost of multiple shipments. This applies the rule of Date Range Optimization Logic. But if the customer wants the items to be shipped when they are available, the shipment optimization is considered and the customer is shipped items1 and 2 at the earliest date 2/6/06 which falls within a single window.

A Delay window can be specified at the item level to account for high ticket items.This window can be set to 0 days which means that the product is never delayed. You can also choose not to delay shipments to be consolidated with future inventory since information (dates) about future inventory is often not accurate. This ensures that onhand and procured inventory is shipped immediately.

- Cost-based scheduling - Cost-based optimization enables the scheduling process to consider landed cost when determing a fulfillment option. Landed cost consists of the following costs:
  - Item Inventory Cost uses the average cost maintained in Selling and Fulfillment Foundation to compute a unit cost of an item at a given node.
  - Handling Cost, both input and outbound handling costs, can be configured per unit, per line, or per shipment. Inbound cost is considered when procurement is selected during scheduling, and is applied at the procure-to node. Outbound handling cost is considered for outbound shipments and is applied at the ship node.
  - Transfer Cost is considered when a procurement option is selected when scheduling. Transfer cost can be configured per unit, per unit weight, per unit distance. Alternatively, a fixed transfer cost can be specified for an individual transfer schedule.
  - Final Leg Cost is used to calculate the cost of a final shipment to a customer. It is usually computed for a specified carrier or carrier service. A user exit is provided to compute the final leg cost.

  When scheduling, the landed cost options are evaluated to determine the least landed cost, which is used as the cost-based option. Landed cost is prorated to reflect landed cost per unit. The priority is configured in the scheduling rules. In addition to the various costs listed previously, the priority can be converted into cost using the cost factor. For example, if a node within a distribution group has a lower priority (higher priority number), the node will contribute more to the cost, compared to a higher priority node.

  To determine the best cost-based optimization type, a higher number of solutions should be evaluated. To evaluate a higher number of solutions, you can increase the MaximumRecords in the promising and scheduling APIs. If date optimization is used, cost-based

scheduling will not be used because the fastest option will be considered first.

Cost-Based Scheduling: An Example

In the following example, landed cost is computed for a given option. Let us assume that a customer orders a DVD player and three DVDs. The DVD player is available at DC1, and the DVDs are available at DC2. In this case, the schedule has two options:

- Option 1 is to ship individual items to the customer as two separate shipments from DC1 and DC2.
- Option 2 is to transfer both items to DC3, and then ship a single shipment to the customer from DC3.

Because option 2 is less expensive, it is selected by the schedule as the cost-based schedule option. The options are described in the following sections.

Option 1

Assuming that all the costs are configured, the landed cost for option 1 is computed as follows:

- In shipment 1, the DVD player is shipped directly to the customer. The following equation is used to calculate landed cost for shipment 1:

  (Total Item Cost + Outbound Handling Cost + Final Leg Cost) / Total Units

  If this equation is used, the landed cost for shipment 1 is $120.50, as shown in the following calculation:

  (1 unit * $100 (average cost) + $3 (per shipment) + $17.50 (ground shipping to customer)) / 1 unit = $120.50

- In shipment 2, three DVDs are shipped from DC3 directly to the customer. The following equation is used to calculate the landed cost for shipment 2:

  (Item Cost + Outbound Handling Cost + Final Leg Cost) / Total Units

If this equation is used, the landed cost for shipment 2 is $14.33, as shown in the following calculation:

(3 units * $10 (average cost) + $3 (per shipment) + $10 (ground shipping to customer)) / 3 units = $14.33

The calculations for shipment 1 and shipment 2 are added together in order to determine the landed cost to fulfill option 1, which is $124.83.

Option 2

Assuming that all the costs are configured, the landed cost for option 2 is computed as follows:

- In shipment 1, the DVD player is shipped from DC1 to DC3. The following equation is used to calculate the landed cost for shipment 1:

  (Total Item Cost + Outbound Handling Cost) / Total Units

  If this equation is used, the landed cost for shipment 2 is $106, as shown in the following calculation:

  (1 Unit * $100 (average cost) + $3 (per shipment) + $2 (fixed transfer cost between DC1 and DC3) + $1 (per shipment at DC3)) / 1 Unit = $106

- In shipment 2, the DVDs are shipped from DC2 to DC3. The following equation is used to calculate the landed cost for shipment 2:

  (Total Item Cost + Outbound Handling Cost + Transfer Cost + Inbound Handling Cost) / Total Units

  If this equation is used, the landed cost for shipment 2 is $11.67, as shown in the following calculation:

  (3 Units * $10 (average cost) + $3 (per shipment) + $1 (fixed transfer cost between DC2 and DC3) + $1 (per shipment at DC3)) / 3 Units = $11.67

- In shipment 3, the DVD player and DVDs are shipped from DC3 to the customer. The following equation is used to calculate the landed cost for shipment 3:

  (No Inventory Cost (items/are procured) + Outbound Handling Cost + Final Leg Cost) / 4 Units

If this equation is used, the landed cost for shipment 3 is $3.33, as shown in the following calculation:

($3 (per shipment) + $7 (ground shipping to customer from DC3)) / 3 Units = $3.33

The calculations for shipment 1, shipment 2, and shipment 3 are added together in order to determine the landed cost to fulfill option 2, which is $121.

Ship Complete Order

This parameter ensures that all product lines in the promising inquiry request are either completely scheduled or not scheduled at all. Lines could, however, be sourced from different shipping locations.

Ship Order from Single Ship Node

This parameter ensures that all product lines in the promising inquiry request are either completely scheduled or not scheduled at all. It also ensures that the complete request is sourced from a single node on a single date. This is a super set of "ship complete order" and when this parameter is set, a "ship complete" is assumed.

Ship Complete Line

This parameter ensures that every product line on an individual line basis is either completely sourced or not sourced at all. However, lines could be sourced from different shipping locations. The difference between this rule and the "ship complete order" rule is that this rule does not enforce that all lines of the request are completely sourced. One particular line can be sourced while another line of the same request could be backordered.

Ship Line from Single Ship Node

This parameter ensures that every product line in the inquiry request is either completely scheduled or not scheduled at all. It also ensures that each individual line is sourced from a single node on a single date. This is a super set of "ship complete line" and when this parameter is set, a "ship complete line" is assumed. However, this rule does not enforce that all lines are shipped from the same node. A particular line may be completely shipped from node 1 while another line could be completely shipped from node 2.

9.2.1.4.6 Other Constraints for Sourcing

The following additional constraints can also be set for product sourcing.

Shipping together a set of product lines

There are situations when a subset of order lines in a single order must be shipped together from the same node and at the same time. Selling and Fulfillment Foundation provides the capability to create a "ship together group of order lines".

A fill quantity can be specified for each line. At a minimum, fill quantity must be available for all of the lines that are part of the ship together group. All lines that are part of this group are shipped from the same node at the same time.

There is no configuration needed for this. On each order transaction, a ship together group can be associated by using setting the "Ship together dependency" parameter. Select a set of lines that need to be shipped together and create the dependency group. Selling and Fulfillment Foundation promising functions ensure that these lines are scheduled to ship together.

> **Note:** This constraint can be specified at line subset level only if an order exists in the system. When making an inquiry without an order, you can specify this constraint at the inquiry level and Selling and Fulfillment Foundation ensures that all the lines in the request are shipped together.

Deliver together a set of lines

The key difference between this constraint and the "ship together" constraint is that with this constraint individual lines that are part of the dependency group can be shipped from different locations, but Selling and Fulfillment Foundation ensures that either all the lines that are part of the dependency group are scheduled for delivery or none of them are. A "merge node" can be specified to consolidate these lines at a location so that it can be ensured that these lines are delivered together. When an order release is made, "merge node" is applied to the release. The shipment consolidation process also ensures that "merge node" is applied to the shipment. The shipping warehouse should make sure that

shipments are made to this "merge node" instead of the actual ship-to location.

There is no configuration needed for this. On each order transaction, a deliver together group can be specified using the "Deliver together dependency" parameter. Select a set of lines that need to be delivered together and create the dependency group. Selling and Fulfillment Foundation promising functions ensure that these lines are scheduled to deliver together.

> **Note:** This constraint can be specified at line subset level only if an order exists in the system. When making an inquiry without an order, you can specify this constraint at the inquiry level, and Selling and Fulfillment Foundation ensures that all the lines in the request are shipped together.

To ensure that every shipment made for an order line has a specified minimum quantity included in that shipment, a fill quantity can be specified on any request line. Selling and Fulfillment Foundation ensures that at least this quantity is shipped in each shipment made out of a node. If an order line's remaining quantity is less than this fill quantity, it is not scheduled. You can either cancel the remaining quantity or reduce the fill quantity appropriately.

Use Single Node For Line Fulfillment

This rule mandates that an order line is fulfilled from only one node. However, unlike the scheduling rule 'Ship Line From Single Node', this rule does not mandate that lines ship completely from a single node. An order line can be shipped partially and the remaining quantity can be shipped later.

If this rule is enabled, it additional features are made available including:

- Line splitting on partial backorder (for lines with and without pre-defined ship nodes)
- Backordering of a line to the most optimal ship node.

For more information about these parameters, see Section 9.2.7, "Backorder Handling".

9.2.2 Calculating Expected Dates

This section describes the product functionality related to calculations of expected shipment and delivery dates for a given inquiry considering various factors such as:

- Minimum notification time required by the node
- Notification schedule
- Level of Service
- Carrier pickup schedule
- Carrier hold time
- Shipping calendar of the node
- Receiving calendar of the node
- Receipt processing time
- Transit time required for shipment
- Predefined transfer schedules between ship-from and ship-to locations
- Sequencing constraints defined between product and service lines
- Product search window - the period of time after the requested date in which an order line can be scheduled

9.2.2.1 Shipment Date Calculations

This section discusses ship date calculations for "As Soon As Possible" (ASAP) orders and future orders.

ASAP orders are orders that do not have any requested ship or delivery dates specified. These orders could be shipped any time possible. Also, all orders that have a requested ship date or requested delivery date occurring in the past are considered as ASAP orders and treated in this same manner.

Future orders are defined as orders that have at least one of either the requested ship or delivery date defined and the specified date(s) occur in the future.

To arrive at the exact ship and delivery date, system initially calculates a range of ship and delivery dates for the order line. System then refines

the range to an exact date based on various locations it is sourcing the product from.

9.2.2.1.1 Initial Calculations of Date Ranges

The initial date ranges for ASAP and future orders are calculated based on the following criteria:

- ASAP Orders
  - Lower boundary of the ship date is set as current time
  - Upper boundary of the ship date is set to the cancel date if specified. If not specified, it is set as current time + "Allowed shipment delay window", a parameter setup in the scheduling rules.
  - Lower boundary of the delivery date is set as current time
  - Upper boundary of the delivery date is set as the cancel date if specified. The only exception to this is if the requested ship date and the cancel date were specified without specifying any requested delivery date. The cancel date in this case is treated as the upper boundary for the shipping date and not the delivery date. If the cancel date was not specified, this is calculated as the upper boundary of the ship date + 60 days to take care of transit time. When actual dates are calculated, this transit is refined to the actual transit time.
- Future Orders
  - Lower boundary of the ship date is the requested ship date if specified. If not specified it is set as current time.
  - Upper boundary of the ship date is set to the cancel date if specified. If the cancel date is not specified but the requested ship date was specified, it is set at the requested ship date + "Allowed shipment delay window", a parameter setup in the scheduling rules. If the requested ship date was not specified, this is set as the upper boundary of the delivery date.
  - Lower boundary of the delivery date is set as the requested delivery date. If not specified, it is set as the lower boundary of the ship date.
  - Upper boundary of the delivery date is set as the cancel date if specified. The only exception to this is if the requested ship date

and the cancel date were specified without specifying any requested delivery date. The cancel date in this case is treated as the upper boundary for the shipping date and not the delivery date. If the cancel date was not specified, this is calculated as the requested delivery date + "Allowed shipment delay days". If the requested delivery date was not specified, it is set as the upper boundary of the ship date + 60 days to take care of transit time. When the actual dates are calculated, this transit is refined to the actual transit time.

Also note that the shipment delay window is specified in elapsed days. It is typically expected that this is set to be at least more than the maximum notification time required by any node or item. If finer control is required over the last date when an order can be shipped, the cancel date should be used for that finer control. Selling and Fulfillment Foundation defaults the "Allowed shipment delay days" to 30 days.

Table 9–2, "Initial Date Range Calculation Combinations" details the various combinations of requested delivery dates, ship dates and cancel dates and how Selling and Fulfillment Foundation calculates the initial ship and delivery date ranges.

Table 9–2 Initial Date Range Calculation Combinations

Requested Delivery Date	Requested Ship Date	Cancel Date	Initial Ship Start	Initial Ship End	Initial Delivery Start Date	Initial Delivery End Date
Not specified	Not specified	Not specified	Current time	End of day after allowed shipment delay days from the current time	Current time	Initial ship end date + 60 days
Not specified	Not specified	Specified	Current time	Cancel date	Current time	Cancel date
Not specified	Future date	Not specified	Future date specified	End of day after allowed shipment delay days from the future day	Future date specified	Initial ship end + 60 days

Table 9–2 Initial Date Range Calculation Combinations

Requested Delivery Date	Requested Ship Date	Cancel Date	Initial Ship Start	Initial Ship End	Initial Delivery Start Date	Initial Delivery End Date
Not specified	Future date	Specified	Future date specified	Cancel date	Future date specified	Initial ship end + 60 days
Not specified	Past date	Not specified	Current time	End of day after allowed shipment delay days from the current time	Current time	End of day after allowed shipment delay days from current time + 60 day
Not specified	Past date	Specified	Current time	Cancel date	Current time	Cancel date + 60 days
Future date	Not specified	Not specified	Current time	End of day after allowed shipment delay days from the future date	Future date	End of day after allowed shipment delay days from the future date
Future date	Not specified	Specified	Current time	Cancel date	Future date	Cancel date
Future delivery date	Future ship date	Not specified	Future ship date	End of day after allowed shipment delay days from the future delivery date	Future delivery date	End of day after allowed shipment delay days from the future delivery date
Future delivery date	Future ship date	Specified	Future ship date	Cancel date	Future delivery date	Cancel date

Table 9–2 Initial Date Range Calculation Combinations

Requested Delivery Date	Requested Ship Date	Cancel Date	Initial Ship Start	Initial Ship End	Initial Delivery Start Date	Initial Delivery End Date
Future delivery date	Past ship date	Not specified	Current time	End of day after allowed shipment delay days from the future delivery date	Future delivery date	End of day after allowed shipment delay days from the future delivery date
Future delivery date	Past ship date	Specified	Current time	Cancel date	Future delivery date	Cancel date
Past delivery date	Not specified Or Past ship date	Not specified	Current time	End of day after allowed shipment delay days from the current time	Current time	End of day after allowed shipment delay days from the current time
Past delivery date	Not specified Or Past ship date	Specified	Current time	Cancel date	Current time	Cancel date

Initial Date Ranges Example

Example 9–3, "Initial Date Range Calculation" illustrates the initial date ranges used for making sourcing and scheduling decisions

Example 9–3 Initial Date Range Calculation

If you set the following parameters as indicated, the initial date range calculation results are those listed in Table 9–3, "Example Date Range Calculation Results".

- Current time = 9/8/2003 3PM
- Allowed shipment delay days = 30 days (Note that typically this parameter should be set to at least the notification time of any node + maximum sequential number of non-working days)

Table 9–3 Example Date Range Calculation Results

Requested Delivery Date	Requested Ship Date	Cancel date	Initial ship start	Initial ship end	Initial delivery start date	Initial delivery end date
Not specified	Not specified	Not specified	9/8 3PM	10/8 00AM	9/8 3PM.	12/8 00AM
Not specified	Not specified	9/30	9/8 3PM	9/30 00AM	9/8 3PM.	9/30 00AM
Not specified	9/15 2PM	Not specified	9/15 2PM	10/15 00AM	9/15 2PM	12/15 00AM
Not specified	9/15 2PM	9/30	9/15 2PM	9/30 00AM	9/15 2PM	11/30 00AM
Not specified	Past date	Not specified	9/8 3PM	9/9 00 AM	9/8 3PM	11/9 00AM
Not specified	Past date	9/30	9/8 3PM	9/30 00AM	9/8 3PM	11/30 00AM
9/15 2PM	Not specified	Not specified	9/8 3PM	10/15 00AM	9/15 2PM	10/15 00AM
9/15 2PM	Not specified	9/30	9/8 3PM	9/30 00AM	9/15 2PM	9/30 00AM
9/15 2PM	9/12 2PM	Not specified	9/12 2PM	10/12 00AM	9/15 2PM	10/15 00AM
9/15 2PM	9/12 2PM	9/30	9/12 2PM	9/30 00AM	9/15 2PM	9/30 00AM
9/15 2PM	9/6 2PM	Not specified	9/8 3PM	10/15 00AM	9/15 2PM	10/15 00AM
9/15 2PM	9/6 2PM	9/30	9/8 3PM	9/30 00AM	9/15 2PM	9/30 00AM
9/6 2PM	Not specified.	Not specified	9/8 3PM	10/8 00AM	9/8 3PM	10/8 00AM
9/6 2PM	Not specified.	9/30	9/8 3PM	9/30 00AM	9/8 3PM	9/30 00AM

9.2.2.1.2 Shipping Calendar

A node can set up a shipping calendar or inherit the calendars of its primary enterprise. This calendar is used to schedule shipments to ensure that they are scheduled only within the working times of the

node. If a node has no shipping calendar set up, Selling and Fulfillment Foundation assumes "24 hours a day for 7 days a week" operations, and schedules the shipment based on other parameters.

9.2.2.1.3 Minimum Notification Time

Minimum notification time represents the minimum number of business hours it takes to ship an order once it has been scheduled to the node. Some nodes (or external suppliers) may need 2 to 3 days after receiving a shipment advice to ship the order. Another node can ship an order within 2 hours of receiving it. This parameter ensures that when making order promises, expectations are set correctly. This parameter can be specified at both the shipping node and item level. The maximum of the two values calculated against the current time or the next available shift is used to determine the notification time. Combined with the shipping calendar parameter, this allows you to make accurate order promises that can be met.

For example, if the minimum notification time is set to 2 hours and the ship node operates between the hours of 8 to 5PM, an order must be scheduled by 3PM to ship on the same day otherwise it needs to be scheduled to ship the next day.

In cases where inventory information is not available from a supplier to whom you are sourcing the order, this parameter can be used to represent the lead time. The expected shipment date is calculated to ensure that it is at least the equivalent of the minimum processing time ahead of the current time thus ensuring that your promises can be kept.

9.2.2.1.4 Notification Schedule

A node can configure specific days and times that it will receive notification of orders for shipping. For example, a node could specify that it can receive notifications only at 9:00 a.m. Monday through Thursday.

This affects the notification time because, even if the node is capable of shipping two hours after receiving a notification, it cannot receive notification on Fridays. This notification schedule will force the notification to the following Monday at 9:00 a.m.

9.2.2.1.5 Level of Service

You can set up different Levels of Service for your business by first defining Levels of Service at the hub level and then configuring notification times and schedules for Levels of Service at the node level. Selling and Fulfillment Foundation provides the regular Level of Service by default. For example, to set up Levels of Service for a business with regular and premium service, first define the premium Level of Service at the hub level and then configure a 10-hour minimum notification time for regular orders and a 5-hour minimum notification time for premium orders.

Level of Service can be specified at the order header and order line levels. If a Level of Service is requested at a node where Levels of Service are not configured, the notification time of the item and the notification time of the node and schedule are used; however, if Level of Service is configured at the node, only the notification time and schedule of the Level of Service at the node are used.

Selling and Fulfillment Foundation searches the order header/line to determine notification schedules, based on the Level of Service on the order, and then uses the specified notification schedule to calculate the expected ship date. Order lines with different Levels of Service are never part of the same release and bundled items always have the same Level of Service.

Level of Service can also be specified for delivered and provided services. For example, a customer may request a Level of Service for a delivery/installation. In this case, the minimum notification time for the premium service would be less than the minimum notification time for the regular service, so the appointment would be sooner. Selling and Fulfillment Foundation ensures the same Level of Service is assigned to all lines of the work order (product and delivery).

9.2.2.1.6 Carrier Pickup Schedule

A carrier pickup schedule of a node determines the dates and times when carriers pick up shipments from the node. Carriers can have schedules for different seasons, such as the holiday and summer seasons, and schedules can include overrides for days, such as holidays. Selling and Fulfillment Foundation refers to the pickup schedules of the different carriers of the node when calculating shipment dates.

For example, if Ground service picks up at 3 p.m. on Mondays, Wednesdays, and Fridays, an order placed at 12 p.m. Monday will ship at 3 p.m. Monday when using Ground Service. A second order placed at 3:30 p.m. Monday will not ship until 3 p.m. Wednesday. In the case of the second order, a different carrier may pick up the shipment if the carrier can make the delivery sooner than 3 p.m. Wednesday.

The minimum notification time of a node also affects when a carrier picks up a shipment. For example, if the node in the above example requires four hours notification before shipping an order and an order is placed at 12 p.m. Monday, the order will ship at 3 p.m. Wednesday using Ground service. Another order placed at 3:30 p.m. Monday will also ship at 3 p.m. Wednesday.

9.2.2.1.7 Carrier Hold Time

The carrier hold time determines the number of days a service carrier holds deliveries that cannot be delivered on a valid delivery day. This parameter ensures that the correct shipment date is set for carriers that do not hold deliveries for more than a specific number of days.

For example, Selling and Fulfillment Foundation will not set Saturday as a ship date for a carrier that holds deliveries for only one day and does not deliver on Saturdays and Sundays. However, Sunday can be set as a ship date in this example because the carrier will hold the delivery on Sunday and make the delivery on Monday.

9.2.2.1.8 Use End of Shift as Shipping Time

This parameter is used to calculate the time component of the shipping date. If product is expected to ship on a particular day, the time can be chosen as the "end of next shift" time for that day by setting this parameter to "Y". If this parameter is set to "N", the time component is set as "next working" time.

Example 9–4, "End of Shift Shipping Time Example 1" and Example 9–5, "End of Shift Shipping Time Example 2" illustrate the usage of these settings for calculating the correct scheduling date.

Example 9–4 End of Shift Shipping Time Example 1

Given the following scenario, an order is scheduled to ship as indicated in Table 9–4, "Scheduled to Ship Time (example 1)".

- Your Seattle distribution center works 6 days a week with Sunday as an off day.
- On Saturdays the working hours are between 8 AM and 12 PM and all other days the working hours are specified in two shifts, 8 AM to 4 PM and 4 PM to 8 PM.
- The Minimum Notification Time = 2 hours
- Product availability is "on-hand"
- The "Request to Ship Order" is "as soon as possible".

Table 9–4 Scheduled to Ship Time (example 1)

If Current Time is...	and "Use End of Shift" is set to...	the order will be scheduled to ship...
Friday 1 PM	Yes	the same day at 4 PM
Friday 1 PM	No	the same day at 3 PM
Friday 3 PM	Yes	the same day at 8 PM
Friday 3 PM	No	the same day at 5 PM
Friday 6:01 PM	Yes	Saturday at 12 PM
Friday 6:01 PM	No	Saturday at 8:01 AM
Saturday 11:59 PM	Yes	Monday at 4 PM
Saturday 11:59 PM	No	Monday at 9:59 AM
Monday 7 AM	Yes	Monday 4 PM
Monday 7 AM	No	Monday 10 AM

Example 9–5 End of Shift Shipping Time Example 2

Given the following scenario an order is scheduled to ship as indicated in Table 9–5, "Scheduled to Ship Time (example 2)".

- Your Washington distribution center works 5 days a week with Saturdays and Sunday as non-working days.

- All working hours are between 8 AM to 5 PM
- The Minimum Notification time = 3 days (72 hours)
- Product availability is "on-hand"
- The "Request to Ship Order" is "as soon as possible".

Table 9–5 Scheduled to Ship Time (example 2)

If Current Time is...	and "Use End of Shift" is set to...	the order will be scheduled to ship...
Friday 1 PM	Yes	Wednesday 5 PM
Friday 1 PM	No	Wednesday 1 PM
Monday 1 PM	Yes	Thursday 5 PM
Monday 1 PM	No	Thursday 1 PM

9.2.2.1.9 Using a Pre-Defined Transfer Schedule Between Two Nodes

If the receiving node and the shipping node have a pre-defined transfer schedule, the transfer schedule is used for determining the correct expected shipment date. On a transfer schedule, you can specify the days of the week on which you ship from the shipping node to the receiving node. Selling and Fulfillment Foundation assumes that shipments can be made to the receiving node only on the days of the week specified on the transfer schedule. This constraint is added in addition to the minimum notification time to come up with the correct shipping date. Time calculations are unchanged.

You can also specify the default transit time it takes to ship from the shipping node to the receiving node. This default time can be overridden by transit times specified for particular days of the week.

Example 9–6, "Transfer Shipments Using a Pre-Defined Transfer Schedule" illustrates the expected ship date and time when using a pre-defined transfer schedule.

Example 9–6 Transfer Shipments Using a Pre-Defined Transfer Schedule

Given the following scenario an order is scheduled to ship as indicated in Table 9–6, "Ship Time Using Pre-Defined Transfer Schedule".

- Shipment is being made from your Seattle distribution center to a distribution center in Portland, Oregon.
- Your Seattle distribution center works 6 days a week with Sunday as an off day.
- On Saturdays the working hours are between 8 AM and 12 PM and all other days the working hours are specified in two shifts; 8 AM to 4 PM and 4 PM to 8 PM.
- The Minimum Notification Time = 2 hours
- A transfer schedule is defined between the Seattle and Portland distribution centers so that shipments are made from the Seattle distribution center to the Portland distribution center every Monday and Thursday.
- Product availability is "on-hand"
- The "Request to Ship Order" is "as soon as possible".

Table 9–6 Ship Time Using Pre-Defined Transfer Schedule

If Current Time is...	and "Use End of Shift" is set to...	the order will be scheduled to ship...
Friday 1 PM	Yes	the upcoming Monday at 4 PM
Friday 1 PM	No	the upcoming Monday at 8 AM
Monday 5 PM	Yes	the same day at 8 PM
Monday 5 PM	No	the same day at 7 PM
Monday 7 PM	Yes	Wednesday 8 PM
Monday 7 PM	No	Wednesday 8 AM

9.2.2.1.10 Requested Date Synchronization

The requested dates is synchronized with requested ship date, delivery date, cancel dates. The dates can be synchronized at the Order header or at the Order Line. Order dates (yfs_order_dates) at header level are effected by the dates on order header record and also by the schedule records for the whole order. Order dates (yfs_order_dates) at line level are effected by the dates on order line record and also by the schedule records for that line.

If the order date table is modified at the order line, then the dates on the order line is synchronized as:

- OrderLine.RequestShipDate=MinShipDate.Requested
- OrderLine.RequestedDeliveryDate=MinDeliveryDate.Requested
- OrderLine.CancelDate=MaxDeliveryDate.Requested.

9.2.2.2 Delivery Date Calculations

Delivery dates are calculated once the ship dates are known. Following factors are taken into account when calculating delivery dates

- Transit time between the ship from and ship to locations.
- Pre defined transfer schedules between two locations
- Days-of-week delivery can be made based on the service selected. For example, Ground service may make deliveries only Monday-Friday whereas Saturday delivery makes deliveries on Saturdays also. Express deliveries can be made all 7 days of the week.
- Days-of-week delivery can be made based on the different delivery and transfer schedules of the service. For example, Ground service may make deliveries on Monday and Friday during its regular schedule and Monday through Sunday during its holiday schedule.

9.2.2.2.1 Initial Delivery Date Range

Selling and Fulfillment Foundation calculates an initial delivery date range when the product can be delivered based on the customer requested dates.

The lower boundary of the initial delivery date range is set as the current time for ASAP orders and requested delivery time for future orders.

The upper boundary of the range is set as the lower boundary + "Max Allowed shipment delay days" from the associated Scheduling rule.

9.2.2.2.2 Determining Transit Time

Transit time is defined as the time required to deliver the product to its ship-to location once it has been shipped from the shipping location.

If the two locations (shipping and receiving) are defined as transfer nodes, Selling and Fulfillment Foundation looks up the transfer schedule setup between the two locations. If such a setup exists, the transit time is picked up from the transfer schedule.

If a transfer schedule setup does *not* exist between the shipping and receiving locations, the transit time is calculated either through a user exit or by using the Selling and Fulfillment Foundation default logic.

- A `getDeliveryLeadTime` user exit can be used to determine the transit time. This user exit is invoked whenever Selling and Fulfillment Foundation needs to calculate the transit time. If you do not implement the user exit, you can set up Selling and Fulfillment Foundation to work in one of the following two modes for transit time calculation
- If using the Selling and Fulfillment Foundation default logic, the transit time is calculated as:

  ```
  A fixed unit of time +  (the distance between ship-to and shipping
  location/average distance per day). Distance is calculated using the
  longitude and latitude definitions of the two zip codes involved.
  ```

  Note: If the zip codes of the locations (ship to and shipping) are not provided, the city, state, and country information of the locations is used to calculate the distance. In situations in which a city has multiple zip codes, the first zip code returned from the database is used to estimate distance.

- The two parameters "fixed unit of time" and "average distance/day" are picked up from different configurations based on another

enterprise-level configuration. Every Enterprise can specify the "Transit time calculation mode" parameter as one of the following.

- Basic mode - transit time calculation does not consider the carrier and service being used for the transit time. The two parameters are used as configured at the enterprise level.
- Advanced mode - transit time calculation is done based on carrier and carrier service combination level setup. The two parameters are used as configured at the combination level. If a carrier and carrier service combination data record does not exist in the database, Selling and Fulfillment Foundation automatically switches to basic mode.
- Based on Service mode - transit time calculation parameters are picked up from the Service record. The Carrier does not play a role in this mode. If no setup exists for the service, Selling and Fulfillment Foundation automatically defaults to Basic mode.

9.2.2.2.3 Considering Days-of-Week Delivery at Service Level

You can specify at each service level the specific days of the week on which a delivery can be made. For example, an express service can be set up to allow deliveries to be made all seven days of the week, but a regular service can be set up to allow deliveries to be made only Monday through Friday.

You can also specify at each service level the specific days of the week on which the carrier transfers shipments. The transfer schedule of the carrier can affect the transit time of a shipment. For example, if the calculated transit time for a Friday shipment is 48 hours and the carrier does not transfer on the weekends, the shipment will not be delivered until Tuesday.

When calculating the delivery date given to a ship date, the delivery date is calculated as the next date that occurs on or after the ship date plus the transit-transfer time. The delivery date should be a valid day of the week for the service chosen.

For example, the transit time is calculated as 48 hours between two locations, "Ground" service is chosen to make the delivery and is configured to make deliveries Monday through Friday. The service transfers shipments Monday through Sunday. In this example, the delivery date is Monday for any Thursday or Friday shipments.

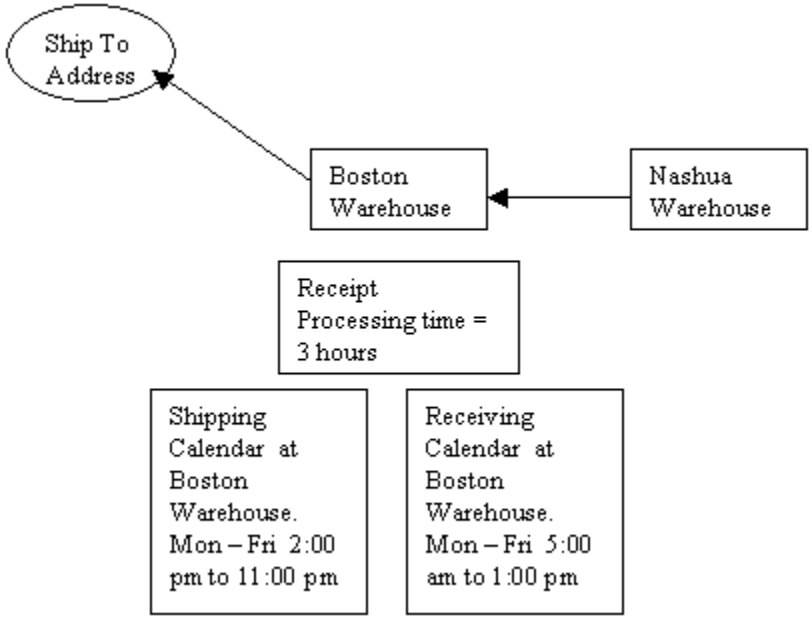
However, if the service is configured to transfer shipments only Monday through Friday, the delivery date is Monday for Thursday shipments and Tuesday for Friday shipments.

9.2.2.3 Calculating Expected Ship Date for Product Procurement

In addition to the previously described date calculations, the Expected Ship Date (ESD) of the order when procuring product takes into consideration the procuring nodes shipping and receiving calendars as well as the nodes receipt processing time.

For example (illustrated in Figure 9–6), an item is being procured from the Nashua warehouse to the Boston warehouse with the criteria.

Figure 9–6 Expected Ship Date Example



- The Expected Delivery Date (EDD) at Boston warehouse is 10/29/03 1:00 pm.

- The EDD to the Boston warehouse is calculated based on previously described dates calculations.
- The ESD on the order is calculated as follows:
  - The Expected Receiving Date at the Boston Warehouse is 10/30/03 5:00 am. (Based on the Receiving Calendar; the Boston warehouse receives Monday through Friday between 5 am and 1 pm).
  - The Expected Receipt (complete) Date at the Boston Warehouse is 10/30/03 8:00 am. (10/30/03 5:00 am + 3 hours receipt processing time). The receipt processing time is added based upon the working hours of the receiving calendar of the Boston warehouse.
  - The Expected Shipping Date from the Boston warehouse is 10/30/03 2:00 pm (Based on Shipping Calendar and assuming 'End of Shift ' is not selected; the Boston warehouse ships Monday through Friday between 2 pm and 1 pm).
  - The ESD on the order is calculated as 10/30/03 2:00pm.

Note: Expected Receiving Date and Expected Receipt Date are not stored in the database, they are used purely for the purpose of this example.

9.2.2.4 Calculating Expected Ship Date for Forwarding

The Expected Ship Date is calculated based on the forwarding node and the shipping node. As explained in the above example, (Section 9.2.2.3, "Calculating Expected Ship Date for Product Procurement") the date calculation remains the same as for product procurement with changes only in the calculation of the Expected Receipt Date. In this case, the Boston Warehouse is considered as the forwarding node and the receipt processing time for forwarding is 0.5 hrs. Therefore, the Expected Receipt (complete) Date at the Boston Warehouse for forwarding is 10/30/03 at 5:30 am (10/30/03 5:00 am + 0.5 hours receipt processing time for forwarding).

9.2.3 Date Synchronization

Dates are synchronized when there is a change in the date for the order line or the order header, and when a rule synchronize dates between master order dates and dates on order line and schedules is configured. The following four order dates are created for both the order line and the order header based on the requested and expected ship dates.

- MinShipDate
- MaxShipDate
- MinDeliveryDate
- MaxDeliveryDate

Every order date has an expected date, requested date, actual date, and committed date. These order dates are specified at either the order header or at the order line level. Requested dates for the order dates are synchronized with requested ship date, requested delivery date, and requested cancel date.

Expected order dates at the line level are synchronized with the expected shipment date, and the expected delivery date of the order line schedule. Expected Order Dates at the order header are synchronized with the expected shipment date and expected delivery date of all the order lines.

Order dates at header level are effected by the dates on the order header record and also by the schedule records for the whole order. Order dates at line level are effected by the dates on the order line record and also by the schedule records for that line.

9.2.3.1 Requested Dates of

The requested dates of the order are synchronized with the requested ship date, delivery date, and cancel date. Requested Dates are synchronized when the createOrder and the changeOrder APIs are called.

If the requested order date table is modified at the order line, then the dates on the order line are synchronized as:

- OrderLine.RequestShipDate=MinShipDate.Requested
- OrderLine.RequestedDeliveryDate=MinDeliveryDate.Requested
- OrderLine.CancelDate=MaxDeliveryDate.Requested

If the requested ship date, requested delivery date, or requested cancel dates are modified at the line level, then the requested order dates are synchronized as:

MinShipDate.RequestedDate = RequestedShipDate

MaxShipDate.RequestedDate = RequestedCancelDate

MinDeliveryDate.RequestedDate= RequestedDeliveryDate

MaxDeliveryDate.RequestedDate = RequestedCancelDate.

The synchronization for the order header follows the similar logic.

9.2.3.2 Expected Dates

The expected dates of the order are synchronized with order schedules. The expected dates are synchronized when an order line schedule is created or modified. When there is a change in the Order Schedule, the ship dates and the delivery dates are synchronized as:

MinShipDate=Min. of all shipdates in the schedules

MaxShipDate=Max. of all shipdates in the schedules

MinDeliveryDate= Min. of all deliverydates in the schedules

MaxDeliveryDate=Max. of all deliverydates in the schedules.

If there is a modification in the expected dates of the order, the order dates are not synchronized with the order schedule.

9.2.4 Impacts of Sourcing Models

So far, sourcing discussions have been related to system determination of the sourcing location and expected dates for determining product availability. Depending on the sourcing model used for sourcing the product, however:

- Inventory may need to be looked up in a different manner
- Shipping notifications may need to be handled differently
- Inventory supply and demand updates may need to be done differently

This section discusses various sourcing models that could be deployed and how Selling and Fulfillment Foundation handles each one of them.

Sourcing Model A

If you ship from stock in one of the locations owned by your organization (or the same legal entity or node marked as "3PL location"), Selling and Fulfillment Foundation handles inventory lookup, shipping notification, and inventory updates as follows:

- Inventory lookup
  - The inventory organization of seller is searched for availability
- Shipping instruction notification
  - Shipping instruction is done through an Order release
- Inventory updates
  - Controlled through the status movement of the order

Sourcing Model B

If you drop ship from a partner. (Shipping location belongs to a different legal entity and not marked as "3PL location"), Selling and Fulfillment Foundation handles inventory lookup, shipping notification, and inventory updates as follows:

The partner organization (vendor) requires a purchase order.

- Inventory lookup
  - If inventory information from the partner is available:
    - The inventory organization of seller is searched for availability if the partner's inventory organization is the same as the seller.
    - If the partner organization keeps inventory externally, a user exit is called to get the inventory information.
    - Note that if the partner organization keeps inventory within Selling and Fulfillment Foundation as part of another inventory organization that inventory is *NOT* accessible.
  - If inventory information from the partner is NOT available:
    - No inventory checks are done. See Section 9.2.2.1.3, "Minimum Notification Time" for calculations of shipping date.

- Shipping instruction notification
  - A "drop ship" chained order is created. This chained order is created in "created" status. Notification can be send to the vendor based on the order that was created at an appropriate step in the pipeline.
- Inventory updates
  - All buyer supply updates are controlled through the parent order only.
  - Seller supply and demand updates are done from parent order.
- The partner organization does not require a separate purchase order

Note that the only inventory model supported for this type of organization is "Inventory maintained as part of Selling and Fulfillment Foundation".

- Inventory lookup
  - Inventory is searched in the inventory organization of seller.
- Shipping instruction notification
  - Shipping instruction is done through an Order release.
- Inventory updates
  - Controlled through the status movement of the order

Sourcing Model C

If you procure inventory from a partner and ship from your own location, Selling and Fulfillment Foundation handles inventory lookup, shipping notification, and inventory updates as follows:

- Inventory lookup
  - If inventory information from partner is available:
    - Inventory is searched in the inventory organization of the seller if the partner's inventory organization is the same as the seller.
    - If the partner organization keeps inventory externally, a user exit is called to get the inventory information.

        * Note that if the partner organization keeps inventory within Selling and Fulfillment Foundation as part of another inventory organization that inventory is *NOT* accessible.
    - If inventory information from the partner is NOT available:
        * No inventory checks are done.

For calculations of shipping dates, see Section 9.2.6.4, "Procurement Purchase Order". Selling and Fulfillment Foundation includes the transit time between the procurement location and shipping location to calculate the final ship date. Selling and Fulfillment Foundation also takes into consideration the receiving calendar and receipt processing time for the final shipping node.

- Shipping instruction notification
    - A "procure to ship" chained order is created immediately at the time of scheduling the order. This chained order is created in "reserved" status.
    - Notification can be send to the vendor based on the order that was created at an appropriate step in the pipeline. A separate release process could be configured that can control notification times through the "advance notification days" setting.
- Inventory updates
    - Inventory updates are done on both orders for both the buyer and the supplier. If the partner organization does not maintain any inventory information, the seller inventory updates for the procurement orders are not be carried out.

Sourcing Model D

If you procure inventory from another location owned by you and ship from another owned location, Selling and Fulfillment Foundation handles inventory lookup, shipping notification, and inventory updates as follows:

- Inventory Lookup
    - Inventory is searched in the inventory organization of the seller.

For calculations of shipping dates, please see Section 9.2.6.3, "Procurement Transfer Order". Selling and Fulfillment Foundation includes the transit time between the procurement location and shipping location to calculate the final ship date.

- Shipping instruction notification
  - A "procure to ship" chained order is created immediately at the time of scheduling the order. This chained order is created in "reserved" status. The document type to be used for this chained order is controlled through a separate document type configuration and, by default, set to 'Transfer order'.
  - Notification can be sent to the node based on the order that was created at an appropriate step in the pipeline. A separate release process should be configured that can control notification times through the "advance notification days" setting.
- Inventory updates
  - Inventory updates are done on both orders for both the buyer and the supplier.

9.2.5 Scheduling Shipment of an Order or Order Line

Selling and Fulfillment Foundation provides the following parameters at various levels to ensure that you have the controls you need to schedule or to NOT schedule an order or order line for shipment until a specific time.

- Status Control
- Earliest Schedule Date

Status Control

This is an order line level control - the order line status should be configured as an available status for the scheduling transaction.

Earliest Schedule Date

This control is used to ensure that future orders are not scheduled too much in advance and thus prevent blocking of available inventory for current or ASAP orders.

The earliest date when an order line can be scheduled is calculated as follows:

- A date can be specified during order line creation. If specified, that date is used as the earliest schedule date.

- If a date was not specified, Selling and Fulfillment Foundation calculates the eligible schedule date as:
  - The lowest of the following values to determine the maximum number of days to schedule before
    - Scheduling rule level parameter "Schedule lead time"
    - Node level parameter "Max days to schedule before". This parameter is used only if node was pre-specified on the order line. If node is not pre-specified, the sourcing step re-determines the earliest schedule date and can decide *not* to schedule the order line at that time.
  - Selling and Fulfillment Foundation then calculates this date as the "ship date" of the order line - the maximum days to schedule before as determined earlier. The requested ship date specified on the order is used as the "ship date". If no specific requested ship date was specified, Selling and Fulfillment Foundation calculates an approximate ship date based on the requested delivery date and the best guess transit time calculation. Note that the approximate ship date calculations may be inaccurate as the actual node from where shipment is being made may not be known. You should ensure that the "Max days to schedule" parameter is set in such a way that the scheduling operation is performed on the order well in time.

Example 9–7 Scheduling Shipment of an Order Line

Given the following scenario the earliest schedule date is calculated as indicated in Table 9–7, "Earliest Schedule Date Calculation".

- You are creating an order line on June 29th 2003 with product as ITEM1.
- You have pre-specified the node on the line as NODE1.

  Your parameters are set as
  - Scheduling rules - Schedule lead time = 30 days
  - NODE1 - Max Schedule days = 10 days

Table 9–7 Earliest Schedule Date Calculation

If Requested Ship Date is...	the Earliest Schedule Date is calculated as...
July 15th 2003	July 5th 2003 and the order is not scheduled until that time.
June 30th 2003	June 29th 2003 and the order is ready for scheduling from this control's perspective.
not specified but a requested delivery date is specified as July 15th, 2003 and the initial transit time was calculated as 3 days	July 2nd 2003. This is because the "ship date" is calculated as July 12th 2003 and hence the July 2nd calculation.
not specified and no requested delivery date is specified	June 29th, 2003.

In each of these instances, order lines are considered for scheduling only after the date calculated. If you want to schedule your order lines as soon as they are created, you should set the maximum scheduling days to a high value such as 999.

9.2.5.1 Skipping Scheduling after the Sourcing Decision

In situations when the shipping location is not known at the time of order creation, the earliest schedule date calculation does not take into account the Node level parameter. When sourcing is done for the order line, the node could be an internal node or an external supplier. You may want to schedule orders immediately if it is an external supplier or may not want to schedule orders to your own node so early and leave the decision open. The scheduling process ensures that even though an order line was considered for scheduling based on initial calculations, it does not schedule the order line if the setup at the node where line is sourced from indicates that it is too early.

Note that in some situations, part of a dependency group may not get scheduled because of this reason as a particular order line in the dependency group may be skipped from scheduling whereas the remaining order lines are scheduled. If this situation is not acceptable, set the "Max schedule days" parameter at the node level to a high value such as 999.

To illustrate this concept, assume that you are creating an order line on June 29th 2003 with your product as ITEM1. You have NOT pre-specified a node on the order line. Your parameters are set as:

- Scheduling rules - Schedule lead time = 30 days
- NODE1 - Max Schedule days = 10 days
- NODE2 - Max Schedule days = 40 days

If the order line had a requested ship date specified as July 31th 2003, The earliest schedule date is calculated as July 1st 2003 and the order is not considered for scheduling until that time. Now, on July 1st 2003 when this order is considered for scheduling. One of the following may result:

- If Selling and Fulfillment Foundation determines the sourcing node as NODE1. Since the "Max schedule days" for NODE1 is set as 10 days, Selling and Fulfillment Foundation does not schedule this order line.
- If Selling and Fulfillment Foundation determines the sourcing node as NODE2. The order line is to be scheduled.

9.2.5.2 Why an Order or Order Line is not Scheduled

The Selling and Fulfillment Foundation scheduling function picks up all order lines that are ready for scheduling based on the controls discussed previously. The scheduling function determines the ship node (if not already determined) and expected dates and schedules the order against the node and date determined. If the scheduling function cannot source the product due to any reason, it backorders the order line (or leaves it in it's current status based on reason of failure). Selling and Fulfillment Foundation updates a "Schedule failure reason" on the order line. This reason explains why the scheduling function did not complete. Possible reasons for scheduling failures are:

- DATE_BEYOND_MAX_SCHEDULE - The ship date (specified or calculated) is in the future and the scheduling rules (Schedule lead time) or the Node parameter (maximum schedule lead days) do not allow scheduling this early. If these parameters are set up incorrectly please correct them and re-run scheduling.
- FILL_QTY_NOT_REMAINING - The order line has fill quantity specified but the remaining quantity is less than the specified fill quantity. The order cannot be scheduled until the fill quantity is adjusted. You can

either cancel the order line for the remaining quantity or adjust the fill quantity if it was specified incorrectly.

- FILL_QTY_NOT_AVAILABLE - The quantity remaining to schedule on the order is more than the fill quantity but it is not available for scheduling yet. The unavailable quantity is in a status that is not ready for scheduling. If the order line needs a manual status change, perform that change, or once the order line is ready, Selling and Fulfillment Foundation schedules the line automatically.
- NO_MORE_LEFT_TO_SCH - There is no more quantity left to schedule for the order line.
- UNAVAIL_FOR_SHIP_COMPLETE - The order line or order scheduling rules are set for line ship complete or order ship complete and all order quantities are not available for scheduling yet.
- SOME_QTY_NOT_AVAIL - For product lines - The order line has a delivery service associated with it but some quantity of the order line is not available for scheduling yet. Selling and Fulfillment Foundation does not schedule the order line until the complete quantity is available for scheduling transaction.
- QTY_NOT_AVL_FOR_SCH_YET - The order line has unscheduled quantity but zero units are available for scheduling.
- APPT_BEFORE_CONSTRAINT_DATE - A delivery service or provided service line has an appointment that occurs before the scheduled delivery date of its pre-sequenced order line.
- APPT_IN_THE_PAST - An appointment on the service line occurs in the past. You need to take the appointment again for the order line to schedule.
- CANCEL_DATE_IN_THE_PAST - The cancel date specified on the order line is in the past. You now need to cancel the order line or adjust the cancel date to an appropriate future date.
- NEEDS_AN_APPT_TO_SCH - This line needs a service appointment before it can be scheduled.
- NO_DEL_SRVC_ATTACHED - The delivery method on the line is set up as "DEL" (Delivery) and there is no delivery service line associated with the order line. Associate a delivery service line.

- PARENT_NOT_COMPLETE - The order line is part of a sequenced line group and it is required that the parent line be delivered before this order line can be scheduled. Wait for the pre-sequenced line to be completed before this line can be scheduled.
- PARENT_NOT_SCHEDULED - The order line is part of a sequenced line group and it is required that the parent line is scheduled before this order line can be scheduled. Look into the reason why the pre-sequenced line was not scheduled and take corrective actions on that line. Once the pre-sequenced line can be scheduled, this line becomes eligible for scheduling.
- CONFIRM_ASSIGNMENT_REJECTED - The scheduling function has found a possible node to ship the order line but the confirm assignment user exit rejected scheduling of this line.
- PARENT_LINE_NOT_READY_TO_SCH - The order line is part of a sequenced line group and it is required that the parent line is scheduled before this order line can be scheduled. The parent line is not ready for scheduling yet, therefore, this line cannot be scheduled. Look into the reason why the pre-sequenced line cannot be scheduled and take corrective actions on that line. Once the pre-sequenced line is ready for scheduling, this line becomes eligible for scheduling.
- DEPENDENCY_CONSTRAINT - The order line is dependent on another line (dependency group or sequenced group) and dependency constraints do not allow scheduling of this line.
- DELIVERY_SERVICE_CONSTRAINT - The delivery service is not available for the line.
- NOT_ENOUGH_PRODUCT_CHOICES - Product inventory is not available for the line. Availability is specific to the date range when product can be shipped. Please check the dates again on the order to see if increasing the "delay window" removes this constraint.
- CAPACITY_NOT_AVAILABLE - The service line does not have enough capacity available for the requested appointment slot.
- DATE_CONSTRAINT - A pre-sequenced line cannot be scheduled before the allowed ship or delivery end date of this order line.
- NO_RESOURCE_POOL_FOUND - No resource pool serves the region or item for the order line. You may have to either cancel the order line or find a similar service that serves that region.

- NO_ADDRESS_SPECIFIED - The order line has no ship-to address specified.
- NO_SOURCING_RULE_DEFINED - The order line does not have any pre-specified ship node or distribution rule and no sourcing rules are defined for the order line.
- NO_PRODUCT_AVAIL - The delivery service line on which this error is shown has delivery capacity but product is not available on the appointment date or time.
- CONFLICTING_CONSTRAINTS/OTHER_CONSTRAINTS - This should happen in rare situations when Selling and Fulfillment Foundation cannot schedule a set of order lines because they have conflicting constraints such as "ship together" but different ship nodes specified on each line.

9.2.5.3 Scheduling an Order or Order Line

Selling and Fulfillment Foundation provides the following mechanisms to schedule an order:

- Use the Schedule agent for scheduling. Based on your pipeline configuration, Selling and Fulfillment Foundation can invoke the schedule agent at an appropriate time and the scheduling function is carried out.
- Use Schedule API to request the scheduling of a particular order. You can request the scheduling of a particular order using this API. Selling and Fulfillment Foundation does essentially the same processing as it does with the scheduling agent except that you are controlling the sequence in which orders are picked up for scheduling.
- Manually schedule the order lines. In this case, you are deciding the node and dates against which the order line should be scheduled. You should use the inventory availability inquiry API before using this API.

9.2.5.4 Scheduling an Order or Order Line that is Reserved

During the scheduling process, it searches for nodes, substitutions, and dates. However, in addition to checking the inventory, the associated reservations are used as pre-reserved inventory. These reservations are used only when they match the reservation parameters. For example:

Order Line: Item1, Qty 5, Node1

Order Line Associated Reservations are:

1. Item2, Qty 3, Node1

2. Item2, Qty 1, Node2

Item2 is preferred substitute for Item1, and Node1 can procure from Node2. While scheduling runs, the only quantity that is available is Qty 1 for Item2 at Node1 or Node2. The remaining quantity is considered as pre-reserved at Node1 and Node2. If the preferred substitute is not configured, then the total of 5 units should be available at Node1 or at Node2, and associated reservations are not used as reservations parameters do not match.

Once schedule runs on the order, it removes all reservations associated to the line and the header. This is regardless of whether reservations were used or not. The release works similarly to schedule and removes all reservations.

9.2.5.5 Scheduling in Reserve Mode

Usually, whenever the schedule runs and it cannot schedule an order for release, the quantity is backordered. Configuration of scheduling rule allows reservation of inventory if it is available but some of the order constraints cannot be met. For instance, an order is placed for 100 units and a fill quantity of 100. If 2 purchase orders for 50 units each are coming 5 days apart, the schedule first reserves 50 units against the first purchase order. Then, within 5 days, the entire line is scheduled once the complete inventory quantity becomes available. Thus, by reserving partially available inventory, the schedule ensures that line is scheduled once the second purchase order arrives. Otherwise, another order with less quantity could consume the first purchase order while the first order would be backordered.

9.2.5.6 Scheduling of an Order Line that is Being Delivered

An order line with its delivery method set to `Delivery` cannot be scheduled unless a service work order has been created for that line.

By default, if a service work order has been created, the line can be scheduled after an appointment has been taken. If the appointment has not been taken and a scheduling attempt is made, the line is dropped in `reserved` status.

However, Selling and Fulfillment Foundation can be configured to allow orders to be scheduled before service work order appointments are taken. For more information, see the *Sterling Distributed Order Management: Configuration Guide*. For more information about service work orders, see Section 10.5, "Service Work Order Types".

> **Note:** If you choose to schedule and release an order at the same time, the delivery order line is never scheduled without a work order appointment being taken for it, even if it is configured as allowed in the Applications Manager.

9.2.5.6.1 Scheduling Bundles

A bundle is represented as multiple order lines with components and parents. An individual line can be both a parent of a bundle as well as a child of another bundle on the order. Each component indicates its parent. The bundle parent indicates how the bundle needs to be scheduled. The possible values for the constraint are Ship Together, Deliver Together, and Independent. This constraint is obtained from the catalog. Kit ratios are enforced during scheduling.

The logic described in this section is considered when performing scheduling activity.

Shipping Constraints

Shipping constraints are obtained from the catalog. If the catalog is not defined, a bundle is shipped together. These constraints are applied to the components at all levels of a bundle.

If the parent indicates "ship independently", no constraints are applied and the components are treated as independent lines.

If the parent indicates "ship together", the components are shipped together from the same node. However, they can be procured from different nodes and then merged at the shipping node to be shipped as a single shipment.

If the parent indicates "deliver together", the expected delivery date drives the ship schedule, and the components have their shipment dates pushed out to meet the same delivery date in ratio.

Out of Ratio

When one or more of the components in a bundle are cancelled, the components fall out of ratio. In this scenario, the constraint 'ship together' is not applied while scheduling the lines that are out of ratio. For example, consider the following:

Ski Package- P1

Skies- C1 (Qty 2)

Poles- C2 (Qty 2)

Helmet- C3 (Qty 1).

Availability:

4C1, 4C2 available at N1, 2C1, 2 C2 available at N2.

The order is created for two Ski Packages and one helmet is manually cancelled. While scheduling, C1 and C2 are completely scheduled against N1, while C3 is backordered. If one helmet is available at N2, it is scheduled against N2, while C1 and C2 are scheduled against N1. Similarly, if the bundle is scheduled against a node, and one of the components is manually backordered, the 'ship together' constraint is not applied. So in the example above, if N2 backorders the second helmet, it schedules C1 and C2 to N1 because it is closer. As a result, all skies and poles ship from N1, while one helmet ships from N2.

Reservations

A few components are reserved when the components cannot be scheduled in ratio and the ratio is necessary. In order to reserve, scheduling runs in two iterations. First, it attempts to schedule all the components and the components that are not scheduled are passed through scheduling again. In the second iteration, lines that have already been scheduled as a part of first iteration are passed through scheduling to be used only for optimizations and ship or deliver together constraints. For example,

Scheduling optimizes on number of shipments:

L1 P1 2 (Ship Together)

L2 C1 Kit Ratio 3 (Total Qty 6 (2*3))

L3 C2 Kit Ratio 2 (Total Qty 4 (2*2))

L4 Item1 (Qty 2)

Inv: Item1 2 N1, C1 - 5 N1, 5 N2

During the first iteration, it schedules L4 against N1, L2 and L3 are backordered. In the second iteration L2 (Qty 5) is scheduled against N1 - not N2. This is because the number of shipments is less when compared to N2.

9.2.6 Notification for Shipping Products

When an order is ready for shipping, one of the following documents is used for communication of this information:

9.2.6.1 Order Release

If shipment is being made from one of your owned nodes or a node not requiring a separate chained document, an order release is sent as the shipping instruction.

Based on the scheduled ship date, Selling and Fulfillment Foundation calculates a notification date. Selling and Fulfillment Foundation then creates an order release on the notification date. The order release is then communicated to the shipping node to carry out the shipping process.

9.2.6.1.1 Calculating Notification Date

The notification date is calculated based on the "advance notification days" parameter and the "maximum working hours" parameter.

Advance notification days are the number of days a ship node needs for notification prior to the expected ship date. This parameter does not take the calendar of the ship node into account. For example, a ship node might require a 2-day advance notification, but if one of these days falls on a non-working holiday, the node will actually need a 3-day notification prior to shipping.

Maximum working hours are the number of working hours a ship node needs for notification prior to the expected ship date, and this takes the calendar of the ship node into account. This parameter addresses the need for ship nodes to factor in the non-work hours when calculating notification dates.

The advance notification day parameter and the maximum working hours parameter can be specified at:

- Ship node level - Advance notification days and maximum working hours. Communication to ship the order to the shipping node or supplier is made at most this many days and this many working hours before the expected shipment date.
- Item Level - Advance notification days and maximum working hours. Communication to ship the order to the shipping node or supplier is made at most this many days before the expected shipment date.
- ATP rule - Advance notification days. This is used only for backward compatibility reasons and is compared with the item setting to pick the maximum of the two. The maximum of the two is used as advance notification time for the item.

Selling and Fulfillment Foundation uses the maximum of all the advance notification days as the advance notification day. Likewise, the maximum of all the working hours is used as the maximum working hour. If a particular ship node needs a longer notification time, you can override it at the ship node level. In some situations, if a particular item needs special handling and a long notification time, you can override that at the item level. Because Selling and Fulfillment Foundation takes the maximum value, timely communication is ensured.

The notification date is calculated as Expected shipment date minus Maximum working hours (including the calendar of the shipping node) minus Advanced notification days. An order release is created on the notification date that can be communicated to the shipping node. Note that Selling and Fulfillment Foundation rechecks the availability of inventory on the notification date before creating the order release. This ensures that if there was a large gap between the scheduling process and the release process, notifications are sent out only if inventory is still available.

9.2.6.2 Drop-Ship Chained Order

If the sourcing function selected an external organization's node that requires you to send a separate purchasing document to be created, Selling and Fulfillment Foundation creates a "drop-ship chained order".

If further control is required on dates when this order should be communicated to the vendor, you can build in processes within the

pipeline to control it. You can also put in an explicit release process for the chained purchasing document.

9.2.6.3 Procurement Transfer Order

A transfer order is created by Selling and Fulfillment Foundation when:

- The final shipping point to the customer is one of your nodes (or a 3PL node or a node owned by same legal entity)
- The shipping node does not have enough stock and needs to be replenished from another node that you own to fulfill the order

Selling and Fulfillment Foundation creates the procurement transfer order at the time of scheduling the original order if:

- The ship node and the node you are procuring from are passed in the order line, or
- Sourcing rules are used and are configured to use ship nodes instead of distribution rules.

The transfer order is created in "reserved" status. You should configure the release agent to release the transfer order. Since an order release is created on the transfer order for communication, the communication date can be controlled as described in Section 9.2.6.1, "Order Release".

9.2.6.4 Procurement Purchase Order

A purchase order is created by Selling and Fulfillment Foundation when:

- The final shipping point to the customer is one of your nodes (or a 3PL node or a node owned by same legal entity)
- The shipping node does not have enough stock and needs to be replenished from an external organization's node.

Selling and Fulfillment Foundation creates the purchase order at the time of scheduling the original order if:

- The ship node and the node you are procuring from are passed in the order line, or
- Sourcing rules are used and are configured to use ship nodes instead of distribution rules.

The purchase order is created in "scheduled" status. You can configure the release agent to release the purchase order or decide to communicate the purchase order directly.

If communicating through the purchase order, you can build controls as required using pipeline configuration.

If an order release is created on the purchase order for communication, communication date can be controlled as described in Section 9.2.6.1, "Order Release".

9.2.7 Backorder Handling

When inventory is not available for an order line, it is backordered. When an order line is backordered it normally needs to wait for scheduling to locate available inventory.

For certain business requirements, backordered lines need to be handled differently.

For example:

- A line requires that a single node is used for fulfillment. If it is partially backordered, it may be required that the backordered quantity is split into a different line and the scheduled quantity moves forward with execution.
- A line on backorder should be backordered to a node based upon sourcing setup (which is the closest to the shipping location or is the highest priority node). In this scenario, it is required that the most optimal ship node is located and used for the backordered line.

Selling and Fulfillment Foundation provides this functionality using additional rules at Enterprise and Document Type combination.

Split On Backorder of Line With Firm Pre-defined Node

This rule applies only when the Use Single Node For Line Fulfillment rule is enabled. With this rule, the system splits any order line with a firm pre-defined node that is being partially unscheduled or backordered). The quantity being unscheduled or backordered is split into a new line.

Split On Backorder of Line Without Firm Predefined Node

This rule applies only when the Use Single Node For Line Fulfillment rule is enabled. With this rule, the system splits any order line without a firm

pre-defined node that is being partially unscheduled or backordered. The portion being unscheduled or backordered is split into a new line.

Backorder Line to the Highest Priority Ship Node

This rule is only considered if Use Single Node for Line Fulfillment is enabled and the ship node is not a firm node. This rule backorders a line to a node which is the best choice for fulfilling this line based upon the sourcing setup. Since inventory is not available at any ship node, the system attempts to find the optimal ship node without performing any inventory checks. If the line is completely backordered, the new node is stamped on the order line.

Note: This rule is only used during the schedule and release process.

9.3 Promising Service Requests

This section discusses the different system components involved in making a successful customer promise for service items such as delivery services or provided services.

9.3.1 Regions and Region Schema

Region and region schema are the building blocks for defining geography in Selling and Fulfillment Foundation.

A region schema represents the complete set of regions that define a specific geographic area.

A region schema consists of a group of hierarchical regions. Each region is itself a set of other regions or a set of zip codes forming that region.

For example, geographical definition of a country like the USA could be:

- A region USA consisting of 50 other regions - the 50 states.
- Each state could consist of a varying number of regions - the counties in the state.
- Each county could consist of cities or towns
- Each city and town could be defined by the set of postal codes it is made up of

Selling and Fulfillment Foundation provides a hierarchical way of defining regions so that aggregate regions could be defined easily and with the least amount of data entry.

Selling and Fulfillment Foundation has a broad functional footprint and sometimes a single way of defining and dividing geography may become very limiting. For example, how territories are defined for shipping may not be the same as required for providing installation services. For reasons like this, Selling and Fulfillment Foundation allows definition of as many Region Schemas as required. Organizations can then associate the right Region Schema for each of the following purposes:

- Shipped Product Region Schema - for purposes of shipping, very broad regions can be defined which divide the country in much fewer regions. These regions can then be associated with shipping nodes from where the product is shipped to the destination.
- Delivery Region Schema - for purposes of providing delivery services, a more granular region definition is desired to the level of cities or towns and in some cases even sections of a city.
- Provided Service Region Schema - as with delivery services, a more granular region definition is desired for this purpose. However, the way the geography is divided might not be the same as it is for a Delivery Region Schema definition.
- Analytics Region Schema - for purposes of Data Warehouse Analytics.
- Selling Region Schema - for purposes of providing different product pricing and entitlements for different regions.

Currently, region schema definition can be done only by the Hub (DEFAULT) organization. Enterprises can select the appropriate schema from the list of schema's that the hub has created. Selling and Fulfillment Foundation provides a basic schema definition for the United States (US) as an optional default setup. You can use this as a starting point for your schema definitions.

While region schemas can only be defined at the Hub level, they can be associated at the resource pool level. If a region schema is associated at the resource pool level, it serves as an override to the region schema at the provider organization level.

Since in any given schema, a large number of regions could be defined, the concept of region level is introduced to allow better manageability. Region level classifies regions into distinct categories to facilitate easier

searches later. Examples of region levels would be "Country", "State", "City", and so forth. Region level also helps to prevent data entry errors. One-time setup defines what "level" can be a child of which "level". This helps in prevention of data entry errors such as adding a "Country" to a "State" inadvertently.

9.3.1.1 How Does Selling and Fulfillment Foundation Identify the Leaf Region for a Given Postal Code?

A region can represent a set of postal code ranges. Selling and Fulfillment Foundation selects the region that includes the required postal code as part of its set. Since multiple regions could include the same postal code, conflict resolution is done in the following manner:

- The region definition having the maximum significant digits in the range is given the highest preference.
  - For example, region R1 has specified a range of 901-902 whereas region R2 has specified a zip code range as 90101-90103. When looking for a postal code of 90101, 90102 or 90103 - R2 is selected as the correct region. When searching for any other postal code between 901-902, R1 is selected as the correct region.
- The region having the least range is given a higher preference.
  - For example, region R3 has specified a range of 90101-90220 whereas region R2 has a zip code range specified as 90101-90103. When looking for postal code of 90101,90102 or 90103 - R2 is selected as the correct region. When searching for any other postal code between 90101-90220, R3 is selected as the correct region
- The starting postal code closest to the required postal code is given a higher preference.
  - For example, region R4 has specified a range of 90099-90102 whereas region R2 has specified a zip code range as 90101-90103. When looking for a postal code of 90101 or 90102, R2 is selected as the correct region. When searching for any other postal code between 90099-90100, R4 is selected as the correct region.

9.3.1.2 Region Match Preferences

Selling and Fulfillment Foundation region matching allows ship to addresses to be matched to regions for application based upon various address fields, depending on the country the region is located within.

The following are the address fields a country can be defined to match by:

- Country
- State
- City
- Zip Code
- Address Line 6

When defining regions, a region level is specified. For example, the region ‘Boston’ would be specified as a ‘City’, while ‘MA’ would be specified as a ‘State’.

Region match preferences enable you to specify which address field to use when matching addresses to regions within a specific country. For example, by specifying ‘City’ as your region match preference for the United States (US), addresses stamped with the country code US are matched by the value of the City field in that address.

9.3.2 Service Slots

When providing any service (Delivery or Provided) an appointment with the customer is required. The appointment is made for the time slot when the service is to be provided.

A capacity organization within Selling and Fulfillment Foundation can define multiple Slot Groups, each containing multiple Service Slots.

Service Slot

A service slot is identified by a start time and an end time. Service promises are made against one of the defined slots.

Slot Group

A Slot Group is identified by a Slot Group ID and is a specified set of service slots. You can associate one slot group to a resource pool.

The ability to define multiple slots and slot groups enables you to take appointments of different granularity for different resource pools. For example, for a resource pool providing delivery service, you can only promise 4-hour time intervals, however you may be able to promise 2-hour time intervals for a resource pool being used for some provided service. Also, sometimes the granularity could differ based on the service provider being used to provide the service.

To illustrate this concept, assume that you provide the following two types of delivery services:

- Curb-side delivery
- White-glove delivery

For curb-side deliveries, you use a third-party delivery service provider who can only make promises to deliver within 4-hour time slots, where as for white-glove deliveries you use your own fleet and can make promises to deliver within 2-hour time slots.

For this example, you would define 2 slot groups each containing the service slots listed in Table 9–8 and Table 9–9.

Table 9–8 Curb Side Delivery Slot Group

Start Time	End Time	Slot Name
8:00 Am	12:00 PM	AM
1:00 PM	5:00 PM	PM
5:00 PM	9:00 PM	Late Evening

Table 9–9 White Glove Slot Group

Start Time	End Time	Slot Name
8:00 Am	10:00 PM	Early AM
10:00 Am	12:00 PM	Late AM
1:00 PM	3:00 PM	Early PM
3:00 PM	5:00 PM	Late PM

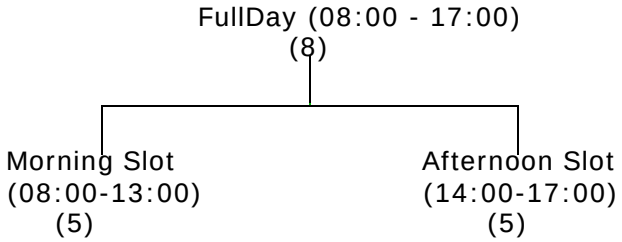
You would associate the first slot group (Table 8-8) with the resource pool providing third-party deliveries and the second slot group (Table 8-9) with the resource pool providing in-house deliveries.

Hierarchical Service Slots

The slots within a slot group can be defined in different levels of hierarchy. The hierarchical slot group may be comprised of parent slots and subsequent child slots. Within a hierarchical slot setup, a parent time slot usually provides the maximum capacity that can be served by all of its child slots. This is to prevent over promising to occur against the actual resources. As illustrated in Figure 9–7, "Hierarchical Slots", if a FullDay slot is considered as the parent slot, then the Morning Slot and the Afternoon Slot within the FullDay slot are the child slots. The child slot itself can be branched to a number of successive child slots. For example, the morning slot can be divided into AM1 and AM2 and the afternoon slot in PM1 and PM2.

The hierarchical slot model allows more granularity and enables appointments to be taken in smaller time intervals. This aids in providing extended customer service and prevents over promising. The parent slot availability restricts the availability of children slots.

Figure 9–7 Hierarchical Slots



(8) - Indicates the total capacity of the parent slot.
(5) - Indicates the individual capacity of child slots.

Customer Slot Preferences

It is possible to define customer appointment slot preference in Selling and Fulfillment Foundation, and associate service time slots with each of them. The associated time slots can either be preferred or mandatory. A customer service representative taking an appointment for a delivery or a service for a customer in the Application Console is able to see which slots are preferred by that customer. If mandatory time slots are used, they are the only ones Selling and Fulfillment Foundation plans an appointment against for that customer. For more information about

defining customer slot preferences, see the *Sterling Distributed Order Management: Configuration Guide*.

9.3.3 Service Items

Service items can be defined within the Selling and Fulfillment Foundation Catalog Management module. Selling and Fulfillment Foundation makes a clear distinction between service items and physical products. Even though there are some core differences in services and physical products, there is a lot of similarity, too. For this reason, both physical products and service items are defined as part of the Selling and Fulfillment Foundation Catalog Management module.

Service items are further divided into the following two broad categories:

- Delivery services are services associated with the delivery of the product. Typical examples would be curb-side delivery or white-glove delivery
- Provided services are services rendered before or after the delivery of the product. Typical examples would be "measurement service" or "installation service".

You can also define an association between physical products and services available for the product. This association can be done at an individual item level or at the item classification level.

Skills for providing these services are maintained at the following levels:

- Service Item
- Service to Product Association
- Service to Item Classification Association

9.3.4 Service Resources

A *service resource* in Selling and Fulfillment Foundation is used to define one or more people that work as a team, and is represented as a single entity that performs provided or delivery services. Each resource has a calendar associated with it by either selecting one of the calendars defined by the ship node or its primary enterprise or using the shipping calendar of a node. This shipping calendar can be defined by the node or inherited from the primary enterprise of the node. Each service resource is associated with a single resource pool.

Skills and regions are defined at the resource pool level, and the resource pool calendar is also used in addition to the calendar of the resource.

This feature allows you to allocate resources at the time of appointment booking. For example, jobs that require multiple visits require the same resource to be booked for multiple days. Also, custom tasks where the skills required to perform the task are much more difficult to model systematically also need resource scheduling to be done. In this case, the scheduler takes into account the unique requirements for the job when selecting the resource.

Additionally, you can associate team members with a service resource in the Applications Manager, as long as those users are defined at the level of the node that owns the service resource. The team members can then be associated with appointments on service work orders. For more information about associating team members with a service resource, see the *Sterling Global Inventory Visibility: Configuration Guide*. For more information about associating team members with a service work order appointment, see the *Sterling Distributed Order Management: User Guide*.

The capacity, consumption, and availability of a service resource or resource pool which maintains capacity at resource level are always calculated and stored in hours regardless of the capacity unit of measure of the resource pool.

9.3.4.1 Capacity Calculation for Service Resources

If a resource pool maintains capacity at the service resource level, the total capacity of the resource pool is the sum of all of the capacities of the resources.

For example, a resource pool (RP1) has the following configuration:

Slots	8am-12pm, 2pm-6pm
Capacity	Working Hours
Mike's Team	9am-12pm, 2pm-6pm (7 hours)
John's Team	10am-12pm, 2pm-6pm (6 hours)
Bob's Team	10am-12pm, 2pm-6pm (6 hours)

The total capacity for RP1 is 19 hours, the sum of the capacities of Mike's Team, John's Team and Bob's Team.

9.3.4.2 Availability Calculation for Service Resources

The maximum availability of a resource pool that can be allocated against is the maximum availability of the services resources maintained in that resource pool.

For example, using the configuration above, the following consumptions exist for Mike's Team in the 8am-12pm slot:

- 2 hour slot based appointment
- 1 hour non slot based appointment

This consumes all of capacity of Mike's Team in the 8am-12pm slot.

	Availability in 8am-12pm Slot	**Availability in 2pm-6pm Slot**
Mike's Team	0 hours	4 hours
John's Team	2 hours	4 hours
Bob's Team	2 hours	4 hours
RP1	4 hours	12 hours

Although the availability of RP1 in the 8am-12pm slot is 4 hours, the maximum availability that can be allocated against is the maximum availability of the service resource, which is 2 hours.

9.3.4.3 Spanning Resource Capacity Across Service Slots

When spanning resource capacity across service slots, the service slot designates the range of time in which the resource is scheduled to arrive and begin performing the services.

Resource pools that maintain capacity at the service resource level and have a service slot group that is not hierarchical can also allow resource capacity to span across slots. In this scenario, service resource availability is determined by considering adjacent shifts.

A capacity organization enables resource capacity slot spanning by setting the applicable resource capacity rule. Once a capacity organization enables resource capacity slot spanning, that capacity organization must specify the maximum number of minutes that a

nonworking shift will be spanned. For example, if you have a resource pool working from 8am to 12pm and 1pm to 5pm, this rule determines whether or not the nonworking time between 12pm and 1pm is allowed to be spanned and for how long. If this rule is set to less than 60 minutes, the slot is not spanned and the slot from 8am to 9am will have 4 hours available for an appointment. If it is set to 60 minutes or greater, it is spanned and the slot from 8am to 9am will have 8 hours available for an appointment. If the resource pool is not working at the beginning of the adjacent slot, the capacity of the adjacent slot is not used. Likewise, if the end of the current slot is nonworking, the capacity of the adjacent slot is not used. The same applies if the slot is consumed at the beginning of the adjacent slot or at the end of the current slot.

Example 9–8, shows the final resource availability when slot spanning is enabled for 60 minutes prior to consumption. In this example, the entire day for the resource pool of one (Joe) begins at 9am and ends at 8pm with a one hour nonworking lunch break.

Example 9–8 Resource Availability with Slot Spanning

Slots	Consumption	Availability
9am-11am	0 hours	10 hours
11am-1pm	0 hours	8 hours
Lunch	0 hours	0 hours
2pm-4pm	0 hours	6 hours
4pm-6pm	0 hours	4 hours
6pm-8pm	0 hours	2 hours

When an appointment is made and the required capacity is consumed, the consumption can be more than the available slot. For example, if a 4-hour appointment is made for Joe to arrive at approximately 2pm, his availability from the 2pm-4pm slot is consumed as well as the availability for the 4pm-6pm slot. Example 9–9 shows the final resource availability when slot spanning is enabled for 60 minutes and a 4-hour appointment has been made.

Example 9–9 Resource Consumption with Slot Spanning

Slots	Consumption	Availability
9am-11am	0 hours	4 hours
11am-1pm	0 hours	2 hours
Lunch	0 hours	0 hours
2pm-4pm	4 hours	0 hours
4pm-6pm	0 hours	0 hours
6pm-8pm	0 hours	2 hours

9.3.5 Resource Pools

A *resource pool* in Selling and Fulfillment Foundation represents the group of individual resources that perform the services. It is the primary mechanism of defining and managing capacity within the Inventory Management module of Selling and Fulfillment Foundation. Note that it is not intended to represent every individual resource that actually performs the service and it is only a means to represent an aggregate service resource. Defining a resource pool gives you the option to define capacity at the resource pool level or at the service resource level.Even when capacity is maintained at the resource pool level, service resources can still be defined for informational purposes. For more information about defining service resources, see Section 9.3.4, "Service Resources".

Every resource pool belongs to one provider organization that owns the resource pool. A resource pool can either provide delivery services or provided services.

Also, a resource pool is associated with a single node. For a delivery service resource pool, this represents the node from where the delivery is made. For a provided service resource pool, this simply represents the node that is responsible for the management of the resource pool capacity.

You can define:

- The regions a resource pool serves and the day of the week as well as time of day when it serves these individual regions
- Standard capacity on a day-of-the-week basis
- Additional capacity

- Supplemental capacity
- Capacity overrides on an exception day basis
- Calendar associated with the resource pool

Selling and Fulfillment Foundation promising functions check capacity availability against the resource pool(s) that match the service and geography requirements of the order line.

Service Skills

Each service request may require certain skills to perform the activity. For example, Washing Machine Installation may require Plumbing and Electrical Skills. Selling and Fulfillment Foundation also ensures that the resource pool selected for a service item should be able to provide all the skills required for the service line.

Additional Fixed Capacity

In the context of a delivery service, certain regions can be considered more difficult to service than others, for certain service types. For instance, you may want to configure Selling and Fulfillment Foundation so that deliveries to suburban areas take an additional 30 minutes compared to deliveries to the city, for complex service types. You can associate additional capacity for any region and service type combination. For more information about configuring additional capacity, see the *Catalog Management: Configuration Guide*.

Multidimensional Capacity

The multidimensional capacity aids in scheduling appointments by restricting the capacity while planning. In addition to primary capacity, additional capacity restrictions can be provided such as:

- Weight
- Volume

Weight

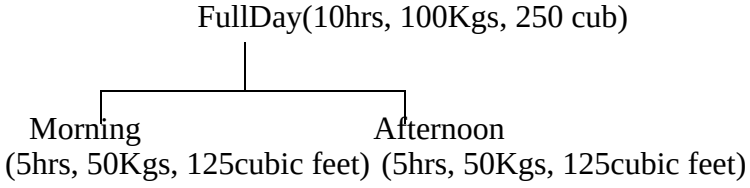
This restricts the weight of products that can be carried on a particular route.

Volume

This restricts the volume of products that can be carried on a particular route.

Capacity is defined and restricted for each time slot. This results in capacity consumed being dependent on the timeslot that is requested. In order to block capacity for the slot, requested capacity in every dimension should be less than available capacity in every dimension for the resource pool. For example, consider a full day slot whose child slots are morning and afternoon slots. The capacity is defined for a full day slot can be restricted by any UOM like 10 hours, 100 kgs, or 250 cubic feet, and the capacity for each child slot can be 5 hours, 50 kgs, and 125 cubic feet. For more details refer Figure 9–8, "Multidimensional Capacity".

Figure 9–8 Multidimensional Capacity



Supplemental Capacity

There can be cases where an enterprise can use supplemental capacity if it is deemed necessary. For example, a company with 10 delivery trucks may be able to use 2 third party trucks if an important client needs products delivered by a certain date. When creating a customer through the Applications Manager, you can specify that by default supplemental capacity is considered for that user. Supplemental capacity can be defined within each standard capacity period, for a given day of the week. You can choose to consider supplemental capacity when taking an appointment for a work order, and you can also choose to consider supplemental capacity when viewing available capacity in the Capacity Console.

Capacity Kept Externally

You can specify a resource pool if capacity information is available at resource pool level. Note that this information can be made available to promising functions either by defining these in the Selling and Fulfillment Foundation Inventory Management module or by flagging the capacity organization as having "capacity kept externally" and providing this information on a real-time basis to the system through a defined user exit.

If capacity information cannot be made available by either of the above mechanisms, you can flag the resource pool to indicate that capacity information is not available. Selling and Fulfillment Foundation treats this similar to "infinite" capacity, but still takes care of the day-of-week and regions served considerations.

When defining a resource pool, you also need to associate it to a single capacity organization. This resource pool can be used only by the Seller organizations whose capacity organization is the same as the resource pool. For more information about capacity organizations, see "Capacity Organization?" and Chapter 3, "Participant Management".

Each resource pool is associated to a slot group. All appointments taken for the resource pool are for the slots defined in this group. All capacity definitions are also for the slots as defined in this slot group.

9.3.6 Promising Delivery Services

Product lines can be associated with delivery service requests. Delivery services are typically provided for products that:

- Are heavy, oversized or fragile and cannot be transported by common carriers
- Require special handling that requires special equipment or personnel

Examples of products that require delivery services are Projection TVs or Washing Machines.

This section discusses product capability related to promising functions for products being delivered using last mile delivery services. For delivering a set of products, a delivery service request must be associated with the product lines being delivered. Selling and Fulfillment Foundation assumes that all product lines must be delivered completely at the same time when scheduling the delivery service. If any part of the

product line is not available (because of inventory, status availability or any other reason), complete delivery is not scheduled.

Typical steps in the promising process for delivery services are:

- Inquiring about an available slot for the delivery and recording a customer appointment based on that availability
- Scheduling the delivery based on the recorded appointment
- Notifying the delivering node to make the delivery

9.3.6.1 Inquiring About an Available Slot

This section discusses the functionality related to finding an available slot for the delivery service.

9.3.6.1.1 Basic Configuration

Some preliminary configuration is related to promising the delivery services that are required with Selling and Fulfillment Foundation. Most of the basic configuration described in this section is defined by the Enterprise organization. To set up your basic configuration for slot availability inquiries, you must set the following parameters as described:

Are delivery sourcing rules defined?

Sourcing rules control the delivery node selection. If you set this parameter with a value of "Yes", the delivering location is determined based on your sourcing rules setup. If this value is set to "No", any delivery location that serves the "shipping region" can be selected by Selling and Fulfillment Foundation. If you have organized your delivery locations such that their delivery regions do not intersect with each other, you can set this configuration as "No" and Selling and Fulfillment Foundation picks up the correct delivering location based on the delivery regions of each node.

If multiple delivery locations could serve the same region and you want to set up sourcing rules for determining the right delivery location, you should set this parameter to "Yes".

Capacity Organization?

Selling and Fulfillment Foundation uses the Capacity organization parameter as a mechanism for separating capacity definition into distinct silos. All resource pools defined in Selling and Fulfillment Foundation

belong to one, and only one, capacity organization. An organization can use only resource pools that are defined within the same capacity organization as that of the organization. This definition allows Selling and Fulfillment Foundation to provide complete isolation for organizations that should not share any resource pools.

You should set up your capacity organization so that it refers to the correct silo. An organization can have one and only one capacity organization and services performed by this organization must be scheduled through a resource pool of the same capacity organization.

Is capacity kept externally?

This parameter controls whether or not the resource pool's capacity is maintained within Selling and Fulfillment Foundation. This parameter is kept at a capacity organization level and if set to "Y", Selling and Fulfillment Foundation assumes that capacity for all resource pools should be fetched in real-time during slot availability checks.

A few of the situations when you would want to set this to "Y" are:

- You are using third-party fleet management tools and do not want to define the slot capacity for each resource pool within Selling and Fulfillment Foundation. During slot availability checks, Selling and Fulfillment Foundation makes a real-time user exit call to find slot availability of the resource pool. You can pass back the availability as read from your fleet management software. Note that even though capacity can be kept externally, resource pools must still be defined in Selling and Fulfillment Foundation.
- You are using third-party service providers who can provide slot availability in real-time. You can define a resource pool representing the third-party provider and make a real-time call to this provider during slot availability checks to get the latest availability picture.

9.3.6.2 Sourcing Rules

A sourcing rule can be created by specifying one or more of the following key parameters:

- Geographical region of the ship-to location or ship-to node
- Fulfillment type
- Seller organization

- Sourcing criteria

You have the flexibility to leave any of the above parameters (except fulfillment type) as void in the sourcing rule and that implies that the sourcing rule is applicable to all values of that parameter.

For each sourcing rule, you can then specify a sequence of node or distribution group to be used for sourcing the product.

As discussed in product sourcing, Selling and Fulfillment Foundation tries to create transfers or purchases if enough stock is not available in the delivering location. Set up for procurement orders is the same as that for shipped products. For more information about procurement configuration, see Section 9.2, "Promising for Products Being Shipped".

9.3.6.3 Finding the Delivering Location

The delivering location is determined based on the following:

- Sourcing rules if the sourcing rules are set up
- Resource pool serving the service region if sourcing rules are not setup

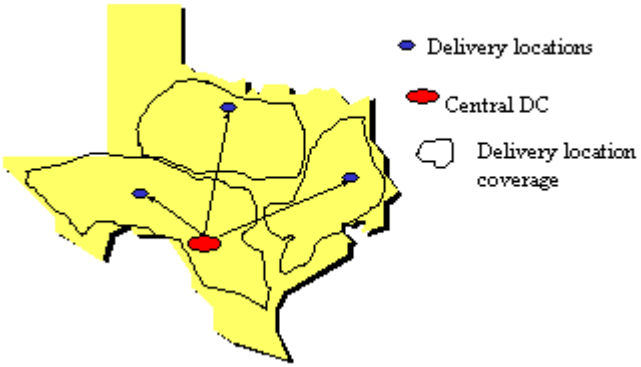
When determining node based on resource pool setup, Selling and Fulfillment Foundation searches for all resource pools that service the delivery region and have the skills needed (if any) to perform the delivery service. All nodes associated with available resource pools are considered for sourcing the product.

Selling and Fulfillment Foundation ensures that:

- The node selected has capacity available for servicing the delivery request.
- The node selected has availability for all the products being delivered as part of this delivery request. Selling and Fulfillment Foundation can be configured to generate automatic transfer or procurement orders from other locations if sufficient stock is not available in the delivering location.

Figure 9–9, "Delivery Locations" illustrates delivery coverage.

Figure 9–9 Delivery Locations



9.3.6.4 Finding an Available Slot

Selling and Fulfillment Foundation considers all the resource pools that are associated with the possible delivering locations and meet the following additional constraints:

- The delivery region must be serviced by the resource pool in order for it to be considered.
- If there are any skills required for performing the delivery, the resource pool provides all those skills.

Selling and Fulfillment Foundation suggests available slots based on:

- Resource pool availability
  - For checking the resource pool availability, Selling and Fulfillment Foundation ensure that:
    - The resource pool has sufficient capacity remaining in the slot being considered. This is considered only if the resource pool has capacity information available. If capacity information is not available, Selling and Fulfillment Foundation consider this as "infinite capacity" and can suggest this slot for recording appointments.

    - The resource pool serves the delivery region on that date based on its "day-of-week" setup for the region. This check is done irrespective of capacity availability.
    - The date is not marked as a non-working day on the resource pool's calendar. The resource pool's calendar could be overridden for the resource pool or the default calendar of the resource pool's node is used.

- Product availability
  - Product availability of all the products being delivered - Selling and Fulfillment Foundation ensures that all the items are available on the first suggested slot.
  - Sequencing constraints - slots are suggested based on sequencing constraints of the order line.
- Minimum service notification time

  A minimum service notification time can be specified at the node level. Any service slots starting within current time and this notification time (in business hours) are considered as unavailable and are not suggested.
- Service search window

  This parameter is set up as part of the scheduling rules. This parameter sets the upper boundary on the date until which service availability is looked up to. Typically you should set this up in the range of 30 to 60 days. Selling and Fulfillment Foundation looks up slot availability within the boundaries defined by this parameter.

  When inquiring for slot availability, Selling and Fulfillment Foundation suggests an array of slots that are available. Since a slot could be served by more than one resource pool, the array represents availability across all resource pools that can serve that region and service item. The actual resource pool to use is not locked in until the order line is scheduled.

9.3.6.5 Capacity Quantity Calculations for Scheduling

Selling and Fulfillment Foundation determines the capacity quantity for which it checks slot availability as "Fixed capacity units to be used for delivery service item + capacity units based on associated product quantity".

Capacity units to be used are based on the product association defined when creating product and service associations in the catalog management system. You can define service capacity to be used / product quantity. Selling and Fulfillment Foundation calculates the variable portion of capacity as (product quantity being ordered) * (service capacity quantity / product quantity). In addition to all this, capacities based on options are added to the total quantity.

Total capacity to check for = Fixed Quantity + Product Quantity Ordered *(service quantity/product quantity) + Options quantity as ordered

> **Note:** If a delivery is already scheduled and an inquiry is being made for additional lines being added to the delivery, fixed capacity and option quantity are not included in the total capacity number.

For more information about the estimated capacity calculations when creating a work order, see Section 10.6.4, "Capacity Calculations for Work Order".

9.3.6.6 Scheduling a Delivery Service

To be able to schedule a delivery service, all product lines associated with the delivery must be available for scheduling. See Section 9.2, "Promising for Products Being Shipped" for more information about product controls.

A service item is moved to Reserved status when scheduled if it does not have a work order or an appointment recorded. During the scheduling process, Selling and Fulfillment Foundation assigns the resource pool that is used for the delivery service. The delivering node selection is done as described in Section 9.3.6.3, "Finding the Delivering Location".

If multiple resource pools are available, Selling and Fulfillment Foundation tries to schedule the delivery service using the resource pool that is *not* marked as the "secondary" resource pool for the ship-to region. If no other resource pool is available, the "secondary" resource pool is used for scheduling.

As part of the scheduling process, capacity consumption is recorded against the resource pool selected.

9.3.7 Other Key Differences from Shipped Products

Selling and Fulfillment Foundation does not perform any transit time calculations for products being delivered. The ship date and delivery date are set to appointment start times when appointments are recorded.

The transfer schedule between the final ship-to location and the delivery location is not used for determining or suggesting slot availability. Slot availability is determined based on resource pool constraints.

9.3.8 Promising Provided Services

You can associate one or more provided services to any product line being ordered. Multiple service lines are associated with a single work order. Each service line on the sales order is represented as a single work order service line. However when taking appointments ONLY resource pools that provide all the service skills are considered. The work orders are also taken against service resources apart from the service slots.

Typical steps in the promising process for provided services are:

- Inquiring about an available slot for the service and recording a customer appointment based on that availability.
- Scheduling the service based on recorded appointment.
- Notifying the delivering node to make the delivery.

Note: Order lines for provided services cannot be used for creating chained orders. Likewise, when the node associated with a provided service order line belongs to a different legal entity the order is scheduled without creating a chained order.

9.3.8.1 Inquiring About an Available Slot

This section discusses the functionality related to finding an available slot for the provided service.

9.3.8.1.1 Basic Configuration

Some preliminary configuration is related to promising is provided services that are required with Selling and Fulfillment Foundation. Most

of the basic configuration described in this section is defined by the Enterprise organization. To set up your basic configuration for slot availability inquiries, you must set the following parameters as described:

Are service sourcing rules defined?

Sourcing rules control the service node selection. You can set up sourcing rules to say whether or not you define sourcing rules at all. If you set up the value of this parameter as "Yes", the servicing location is determined based on your sourcing rules setup. If this value is set to "No", Selling and Fulfillment Foundation selects the servicing location based on the "service region" served by the location.

If resource pools in your capacity organization can be used and you do not typically have overlapping service regions between multiple service providers, this parameter value set to "N" can save you the detailed configuration required for service provider selection.

Set this parameter to "Yes" if you want finer control over provider selection. Selling and Fulfillment Foundation uses your defined sourcing rules and selects the correct provider.

Capacity Organization?

The same capacity organization is used for both delivery and provided services. For more information about this parameter, see "Capacity Organization?" in the delivery service section.

Is Capacity kept externally?

This parameter has the same behavior as defined for delivery services. For more information about this parameter, see "Is capacity kept externally?" in the delivery service section.

9.3.8.2 Distribution Groups

A distribution group provides the facility for you to create a set of nodes or provider organizations. You can then associate this group with a sourcing rule. Selling and Fulfillment Foundation considers all the nodes or organizations that are part of this group and optimizes on various factors to make the final selection. A priority number can be specified against each node or organization that is used for node or provider selection.

If you have multiple nodes or providers and you want Selling and Fulfillment Foundation to select the node based on its build in optimization logic, create such groups and let Selling and Fulfillment Foundation make the final choice of node based on your optimization parameters. If you want to sequence the node selection in a pre-defined and fixed manner use the "sequencing" feature of the sourcing rules setup so that Selling and Fulfillment Foundation can look up nodes in a fixed order.

9.3.8.3 Sourcing Rules

Sourcing rules are defined similar to sourcing rules for products being shipped. Except for the difference that Product Class and Item Classification parameters are not available for setting up the sourcing rule, the rest of the sourcing rule configuration is the same as that described in Section 9.2.1.4.3, "Sourcing Rules" for products being shipped.

9.3.8.4 Finding the Servicing Location

The servicing location is determined based on:

- Sourcing rules if the sourcing rules are set up
- The resource pool serving the service region if sourcing rules are not setup

When determining the node based on resource pool setup, Selling and Fulfillment Foundation searches for all the resource pool's that service the region. All the nodes associated with available resource pools are considered for sourcing the service.

Selling and Fulfillment Foundation ensures that the node selected has the capacity available for servicing the service request.

9.3.8.5 Finding an Available Slot

Selling and Fulfillment Foundation considers all the resource pools that are associated with the possible servicing locations and meet following additional constraints:

- The serviced region must be serviced by the resource pool for it to be considered.

- If there are any skills required for performing the delivery, the resource pool provides all those skills.

Selling and Fulfillment Foundation suggests available slots based on:

- Resource pool availability
  - For checking the resource pool availability Selling and Fulfillment Foundation ensures that:
    - The resource pool has sufficient capacity remaining in the slot being considered. This is considered only if the resource pool has capacity information available. If capacity information is not available, Selling and Fulfillment Foundation considers this as "infinite capacity" and can suggest this slot for recording appointments.
    - The resource pool serves the delivery region on that date based on its "day-of-week" setup for the region. This check is done irrespective of capacity availability.
    - The date is not marked as a non-working day in resource pool's calendar. The resource pool's calendar could be overridden for the resource pool or the default calendar of the resource pool's node is used.
- Minimum service notification time

  A minimum service notification time can be specified at the node level. Any service slots starting within current time and this notification time (in business hours) are considered as unavailable and are not suggested.
- Service search window

  This parameter is set up as part of the scheduling rules. This parameter sets the upper boundary on the date until which service availability is looked up to. Typically you should set this up in the range of 30 to 60 days. Selling and Fulfillment Foundation looks up slot availability within the boundaries defined by this parameter.

  When inquiring for slot availability, Selling and Fulfillment Foundation suggests an array of slots that are available. Since a slot could be served by more than one resource pool, the array represents availability across all resource pools that can serve that region and service item. The actual resource pool to use is not locked in until the order line is scheduled.

9.3.8.6 Scheduling a Provided Service

Controls to determine when a provided service line should be scheduled are similar to those for delivery services. See Section 9.3.6.6, "Scheduling a Delivery Service" for more details.

9.4 Complex Sequencing of Order Lines

Controls are provided so that you can sequence various product deliveries and associated services. You may want to complete certain pre-delivery services before a product is delivered and then perform some post-delivery services ensuring that these are scheduled only after the product has been delivered.

When setting up a product-to-service association, an offset can be specified. A -ve offset denotes that the service must be performed at least offset number of hours *before* the product delivery. A +ve offset denotes that the service must be performed at least offset number of hours *after* the product has been delivered. The hours specified are treated as elapsed hours and not business hours. For this discussion, whenever a service line is sequenced before the product line, it is referred to as a pre-sequenced line, otherwise it is referred to as a post-sequenced line.

Combined with the offset hours, line dependencies explained earlier create a complex sequencing situation where a line should not be scheduled before or after a certain time. Selling and Fulfillment Foundation calculates two constraining dates:

- Cannot complete before date - the date before which the order line should not be scheduled for completion. Completion is denoted by the delivery of product lines and recording of service completion for service lines. This date can be imposed on a line because of any of the following:
  - A pre-sequenced line has been scheduled already. The constraint date represents the last scheduled delivery date of the pre-sequenced line + offset hours specified on the product-to-service association.
  - A pre-sequenced line has a requested delivery data specified. This date represents requested delivery date of pre-sequenced line + offset hours.

  - A pre-sequenced line already has an appointment recorded. This date represents the last appointment time + offset hours.

- Cannot complete after date - The date after which the order line should not be scheduled for completion. Even though Selling and Fulfillment Foundation calculates and shows this date in its user interfaces, this date is not used as a real constraint on the pre-sequenced line. The main reason for this is that in some situations a dead lock can be created between pre-sequenced and post-sequenced lines and no scheduling operations can be performed. To avoid such situations, this constraint is treated as a soft constraint and not imposed on the line. Selling and Fulfillment Foundation highlights lines that do not meet the "Cannot complete after date" constraints. This date can be imposed on a line because of any of the following:

  - A post-sequenced line has a requested cancel date specified. This date represents the requested cancel date of the post-sequenced line - offset hours.

  - A post-sequenced line has an appointment date recorded. This date represents the first appointment date of the post-sequenced line - offset hours.

Selling and Fulfillment Foundation highlights an order line if it has an appointment or delivery schedule that does not meet the above constraints.

Selling and Fulfillment Foundation also calculates an indicator called "Cannot schedule". This indicator denotes that an order line that has this indicator set to "No" cannot be scheduled. Reasons for this are caused by one of the following:

- A pre-sequenced line has not been scheduled yet. All pre-sequenced lines must be scheduled before a post-sequenced line is scheduled.
- When associating product with services "Hold scheduling for completion" was marked as true and a pre-sequenced line has not been completed yet.

Even though a line may be marked as "cannot schedule", Selling and Fulfillment Foundation lets you inquire for the best possible appointment availability. You can also record the desired appointment. However, you cannot schedule the line unless the constraints preventing the line from being scheduled have been removed.

Using the above parameters you can effectively record appointments for lines requiring complex sequencing and then schedule them at appropriate times.

9.5 Item-Based Allocation

Item-based allocation (IBA) is a process that reallocates the uncommitted and committed demands for items of existing orders to more suitable supplies based on user configuration and supply and demand changes in the system.

Selling and Fulfillment Foundation provides two types of item-based allocation options:

- FIFO prioritization, giving orders that are placed first a higher priority than those placed later.
- User-configured prioritization based on order attributes, giving the specified orders higher priority than others

When user-configured IBA priorities are not set, or they are set and completed, Selling and Fulfillment Foundation defaults to system-configured IBA priorities.

This subsequent sections describe both types of IBA.

9.5.1 FIFO IBA

In the default, FIFO IBA, the demands of the impacted orders are reallocated in the FIFO order. This means that the orders that are placed first are promised prior to the placing of orders. For example, the orders shown in Table 9–10, "IBA Example" exist in the system with their corresponding supplies and demands. Assume that it takes one day to ship out orders after a purchase order has been received.

Table 9–10 IBA Example

Purchase Order	Item	Node	Qty	ETA	Sales Order	Item	Node	Qty	Demand Ship Date	Order Date
PO1	Item1	Node1	5	11/15/08	SO1	Item1	Node1	5	11/16/08	11/03/08
PO2	Item1	Node1	5	11/20/08	SO2	Item1	Node1	5	11/21/08	11/05/08
PO3	Item1	Node1	5	11/21/08	SO3	Item1	Node1	5	11/22/08	11/10/08

If the sales order "SO1" is cancelled and FIFO item-based allocation is considered, the "SO2" and "SO3" orders that were placed after "SO1" are reallocated to an earlier date because the supply becomes available on an earlier date as shown in Table 9–11, "FIFO IBA Result":

Table 9–11 FIFO IBA Result

Purchase Order	Item	Node	Qty	ETA	Sales Order	Item	Node	Qty	Demand Ship Date	Order Date
PO1	Item1	Node1	5	11/15/08	SO1	Item1	Node1	0	N/A	11/03/08
PO2	Item1	Node1	5	11/20/08	SO2	Item1	Node1	5	11/16/08	11/05/08
PO3	Item1	Node1	5	11/21/08	SO3	Item1	Node1	5	11/21/08	11/10/08

The Item Based Allocation functionality consists of two processes that should be performed one after the other, as described here:

1. Identifying items to perform Item-based Allocation.

   The process of identifying items is as follows:

   a. Detecting when Item-Based Allocation is required for an item:

      Selling and Fulfillment Foundation detects that the Item Based Allocation process is required for an item at a node when there is no on-hand availability. This occurs when Selling and Fulfillment Foundation is retrieving the availability of an item with the intent to perform an update through promising APIs such as reserveAvailableInventory, modifyWorkOrder, scheduleOrder, and releaseOrder. If the Use Item Based Allocation rule is enabled and there is no on-hand availability at any node, an Item-Based Allocation trigger is inserted (if it does not already exist) into the YFS_IBA_TRIGGER table for the item and node combination with the IBA_REQUIRED flag set to "Y" (only when the item and node have IsItemBasedAllocationAllowed ="Y"). This indicates that the IBA process is required for the item at the indicated node and the system is now allowing the triggering of the item-node combination to be processed by the IBA agent.

   b. Triggering the Item-Based Allocation agent to process an item and a node:

      After Selling and Fulfillment Foundation detects that the Item-Based Allocation process is required for an item at a node (a

record exists in the YFS_IBA_TRIGGER table with IBA_REQUIRED="Y"), Selling and Fulfillment Foundation allows triggering of that item-node combination to be processed by the Item-Based Allocation agent, based on the following occurrences.

- When Supply Changes

  When changes in the supply occur and the supply type has TriggerItemBasedAllocation="Y", Selling and Fulfillment Foundation triggers the inventory item and node to be processed by the IBA agent by updating the IBA_RUN_REQUIRED field to "Y".

- When Certain Demand Changes

  When there is a change in the demand, the inventory item and the node are triggered to be processed by the IBA agent by updating the IBA_RUN_REQUIRED field to "Y" in the YFS_IBA_TRIGGER table.

  When the following changes occur, the inventory item and node are triggered to be processed by the IBA agent by the IBA agent by updating the IBA_RUN_REQUIRED field set to “Y” in the YFS_IBA_TRIGGER table.

  Order cancellation

  Order backordered or backordered from node

  Reservation cancellation

2. Performing the Item Based Allocation process

   Item-Based Allocation process is performed by the Item-Based Allocation agent under the General process type. For each item-node combination that is triggered, the agent finds all the applicable order lines or order line reservations containing the item-node combination and tries to move their uncommitted and committed demands to a more suitable format based on the available supplies, which is in turn based on the user-configured IBA selection rules or the FIFO (First-In-First-Out) IBA selection rules.

   The system then creates new positive order line reservations with the matched supply's first ship date and negative order line reservations for the existing demand ship date. After the orders are processed, they are put on hold to be rescheduled if changes are detected in the

order line reservations of the order. The rescheduling process performs the actual rescheduling of the orders by utilizing the order line reservations created by the Item-Based Allocation process.

Note: A hold type is required to be set up for the change order line reservations modification type so that the order can be placed on hold for scheduling.

9.5.2 User-Configured IBA

In user-configured IBA, rules can be defined to provide sequences and conditions to orders and order lines. A condition can be created, and orders matching this condition will be given priority. For example, in Table 9–12, "IBA Example with Buyer Organization Code", Buyer1, Buyer 2, and Buyer3 are competing for inventory.

Table 9–12 IBA Example with Buyer Organization Code

Purchase Order	Item	Node	Qty	ETA	Sales Order	Item	Node	Qty	Demand Ship Date	Order Date	Buyer Org
PO1	Item1	Node1	5	11/15/08	SO1	Item1	Node1	5	11/16/08	11/03/08	Buyer1
PO2	Item1	Node1	5	11/20/08	SO2	Item1	Node1	5	11/21/08	11/05/08	Buyer2
PO3	Item1	Node1	5	11/21/08	SO3	Item1	Node1	5	11/22/08	11/10/08	Buyer3
PO4	Item1	Node1	5	11/25/08	SO4	Item1	Node1	5	11/26/08	11/15/08	Buyer4

Table 9–13, "User-Configured IBA Result" shows that if Buyer3 is configured for priority, the demand ship date for Buyer3 will change to 11/16/08, giving Buyer3 priority over Buyer2 and Buyer4. Because user-configured priorities are completed, the FIFO priorities take effect, with Buyer2 and Buyer 4 competing for inventory. Buyer2 gets the earlier demand ship date due to its earlier order date.

Table 9–13 User-Configured IBA Result

Purchase Order	Item	Node	Qty	ETA	Sales Order	Item	Node	Qty	Demand Ship Date	Order Date	Buyer Org
PO1	Item1	Node1	5	11/15/08	SO1	Item1	Node1	0	N/A	11/03/08	Buyer1
PO2	Item1	Node1	5	11/20/08	SO2	Item1	Node1	5	11/21/08	11/05/08	Buyer2
PO3	Item1	Node1	5	11/21/08	SO3	Item1	Node1	5	11/16/08	11/10/08	Buyer3
PO4	Item1	Node1	5	11/25/08	SO4	Item1	Node1	5	11/22/08	11/15/08	Buyer4

9.6 Rescheduling

Rescheduling of an order is allowed when the attributes that are important for scheduling are modified. This is enabled only through order holds and the following user configuration:

- The user has to create a hold type
- This hold should prevent ReleaseOrder and CreateChainedOrder Transactions from picking an order
- This hold should be triggered by modification types that impacts scheduling (Order Line Reservation Change, ShipToAddress Change, Node change, and so on)
- This hold should be resolvable by SCHEDULE transaction.

Whenever schedule picks up an order, it checks if the order has any active holds that schedule is configured to resolve. If so, then it confirms that schedule is in reschedule mode. After the schedule ends, it closes all holds that were configured to resolve, unless an order could not be processed.

When there is any modification to the line and line has been scheduled, order line schedules remain unchanged. The line goes on hold and then is be rescheduled. Once the schedule ends, the hold type is removed from the order.

10

Value-Added Services

Value-Added Services (VAS) are performed to meet customer demands. They can be activities performed on a product before the product is delivered to the customer, or a provided service that is performed at the customer site.

Activities can be performed at all types of facilities including manufacturing facilities and flow-through or distribution centers. Some examples of VAS done at the vendor site are:

- A vendor applies a license plate number (LPN) label on a carton based on the manufacturer's specification.
- A manufacturer applies price ticket or SKU labels based on the requirements of a retailer.
- A warehouse builds a pallet using customer requirements.
- A warehouse applies security tags on a class of items before shipping to certain retailers.
- A warehouse assembles components and builds a kit before shipping.

Provided services are usually purchased by the customer for an additional fee. Some examples of activities that are offered as a provided service are:

- Installing a customer’s home theater system.
- Providing maintenance to a furnace as part of a service contract.
- Installing software on a new computer, and configuring the computer to work on a home network.

10.1 Using Value-Added Services

Some examples of how value added services are used within the manufacturing, vendor, or shipping facilities are:

- **VAS for stocking** — a facility performs the VAS operations for stocking periodically on the basis of expected demand for a product. For example, a warehouse ships pens, pencils and erasers separately. In addition, that warehouse may also ship a student package, which consists of 2 pens, 2 pencils and an eraser.
- **VAS for a customer** — this type of VAS operation is performed to meet specific customer requirements, such as special ticketing or security tagging. The Selling and Fulfillment Foundation allows the customization of inventory for a buyer at a node and ensures allocation to that customer. This scenario is referred to as **made-to-customer**, or **buyer compliance**. The customer usually makes repeated purchases of these specialized items. This type of VAS is performed in a warehouse based on the demand forecast to reduce order cycle time.
- **VAS for an order** — this is applicable when:
  - A customer selects a customization when ordering an item, like monogam on a shirt. The Selling and Fulfillment Foundation creates a work order to perform these services in a warehouse and ensures that inventory is allocated only to that order. This is also referred to as **made-to-order**.
  - A customer orders a dynamic physical kit and selects the required components for that item. The Selling and Fulfillment Foundation creates a work order to assemble the kit and ensures that the inventory of the assembled unit is allocated only to that order. This is also referred to as **Built to Order**, or **made-to-order**.
  - Inbound shipments that require packaging. Parts received that require packaging to a specified unit of measure.
- **Breaking kits or Dekitting** — this is performed on kits in inventory that are not required anymore. For example, inventory leftover from a seasonal promotion may be dekitted and the components can be used in other kits or sold individually.
- **Unit of Measure (UOM) Conversions** — a warehouse may create a work order to convert inventory from one UOM to another. For

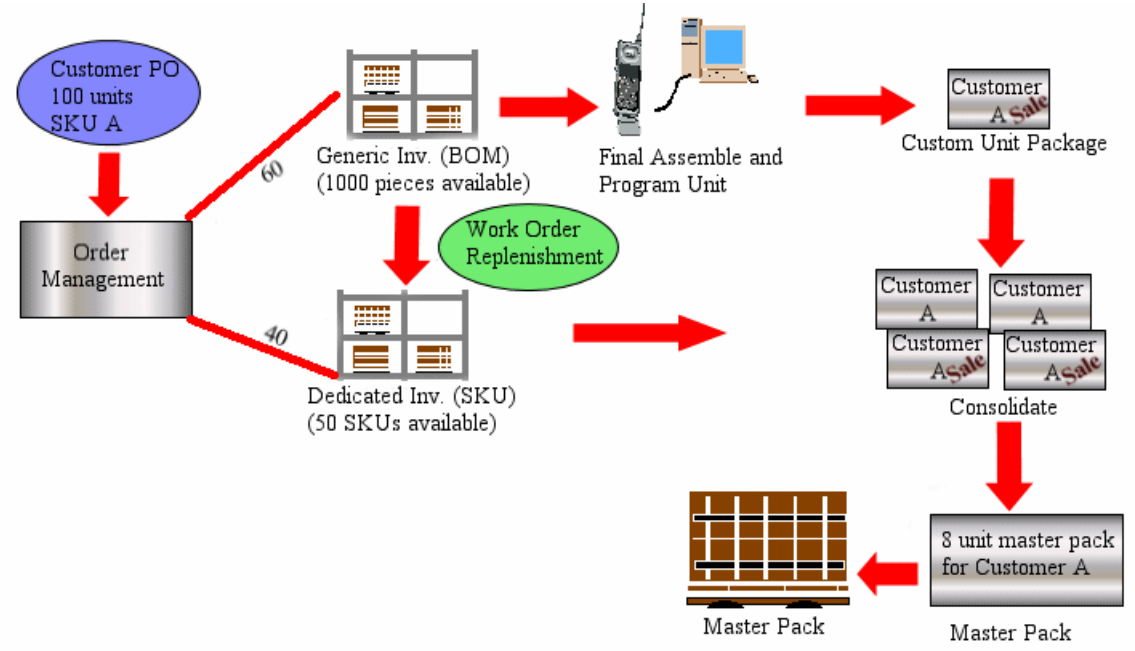
example, a warehouse may track inventory of screws individually as well as in packs of 50. A work order is created to convert single inventory units to packs of 50.

- **Gift Wrap Services** — a ship node may offer gift wrap services on items that are eligible for gift wrap. This flag is enabled for a node during Participant Setup for Sourcing/Scheduling Node Attributes.

10.2 Postponing Item Creation

One approach to providing Value-Added Services at the warehouse or other facility is to delay customization processes until an order is received. This is referred to as 'postponement'. An example of a postponement process is shown in the following figure:

Figure 10–1 Postponement Process



The warehouse stores generic product awaiting the customer demand to customize the product. After receiving an order, additional services are performed on the generic inventory to convert the generic inventory into

customer requested specific inventory. This increases the order cycle time while reducing the risk of unusable inventory.

In the Selling and Fulfillment Foundation, value-added services are executed by creating a work order. A work order is initiated automatically based on a demand, or manually by a user. The Selling and Fulfillment Foundation manages the execution of a work order through retrieval of inventory to the VAS department, recording of completion towards the final inventory, and putaway of the product to appropriate departments.

10.3 Work Order Creation

A work order in the Selling and Fulfillment Foundation captures the activity required to perform a service. A work order is created in the Selling and Fulfillment Foundation by one of the following methods:

- Manual creation - A user initiates a request using the Work Order Console for repackaging, UOM conversions, labeling, or dekitting.
- Automatic creation based on Inventory levels - A request is issued by the inventory monitor in the Selling and Fulfillment Foundation upon minimum or maximum levels being reached for a SKU.
- Automatic creation based on an order - A request issued by a sales order for an item on the order. This typically occurs for **made-to-order** or **made-to-customer** items.

When a work order is created, it consists of one or more of the following types of services:

- Kitting Service - Assembling components for a kit item
- Dekitting Service - Disassembling a kit item. This may be done to acquire an individual component to complete another order.
- Compliance Service - Value-added services that should be performed to supply an item to a specific buyer for **made-to-customer** orders.
- Inventory Change - Converting inventory from one UOM to another.
- Provided Service - Service executed at a customer site.
- Delivery Service - Service executed for product delivery to the customer site.

One or more services can be included in a work order, and have a sequence number assigned which indicates the order in which they

should be performed. Each service can contain one or more service activities, which specifies the category of activity, such as Assemble Components, Apply Logos, or Pack Components. Provided Services are not combined with other services.

10.4 Work Order Hold Types

Work orders may need to be held at one point through their life cycle. For example, certain complex services may require a number of compatible products and services, as well as service tools. A supervisor may be required to verify work orders with such services before appointments can be taken.

The order hold type functionality, described in Section 7.8, "Order Hold Processing", is also available to work orders. The only difference is that a work order cannot be in a draft status, therefore the hold can only be created automatically on work order creation, on resolution of another hold type, or through a modification type.

10.5 Service Work Order Types

The service work order consists of work order service lines, delivery lines and appointments that had to be taken for the work order. The key difference between the service work order and other value-added services is that the services associated with a service work order are usually supplied to the customer at the customer's location, while value-added services are usually performed at the warehouse or other assembly location. Although they differ in this way, many businesses treat provided services as a work order that can either be done by the seller or can be performed by a third party.

A service supervisor can be specified for a service work order. Typically, a supervisor is responsible for managing work orders that originate from a specific set of nodes. If a store supervisor is specified on a service work order, appointments are only allowed to be placed against the resource pools supervised by that user.

The supervisor can be set at the following three levels, in order of priority:

- For a given node and seller organization combination
- For a given resource pool

- For a node

For example, if a given service work order is placed for a given seller organization, the supervisor defined for that seller organization and node combination is chosen over the default supervisor defined for that node.

A service work order is divided into two basic types:

- Provided Service Work Order
- Delivery Service Work Order

10.5.1 Multiple Service Lines on a Single Work Order

You can have multiple service lines on a single work order. Each service line on the sales order is represented as a work order service line.

For example, consider a case where a customer orders a TV, receiver, speakers and installation for each one of the products modeled as separate service lines. If the customer requires everything to be installed together during a single visit, all the three installations can be included on one work order. A single appointment to install all three services is taken. In this case only a resource pool that can provide all three services is considered when taking appointments. Therefore if the customer needs to change the appointment dates later, only a single modification is required.

A service work order with multiple service lines takes on the highest service level based on the service types associated to the service items. Furthermore, service items with the same item group code as the work order are considered for service level determination. For example, if a provided service work order contains a provided service item with a ‘Low’ service level and a delivery service item with a ‘High’ service level, the service level on the work order is ‘Low’. The service level is recomputed every time a service line is added or removed.

10.5.2 Work Order with Service Resources

When a work order is more complex and requires people with different skills, resource pool blocking is not sufficient and taking appointments against service resources is pertinent. A resource is a team that can provide either a provided or a delivery service. For more information about defining service resources, see the *Sterling Global Inventory Visibility: Configuration Guide*.

Whenever a resource pool defines resources, appointments can be taken against the resource based on either calendar dates or time slot. For example, an appointment on the work order for a resource is between 8:30 AM - 5:30 PM.

10.5.3 Multi-Day Work Order

The ability to make multiple appointments for a single work order is helpful if you have a job that spans over a few days. In order to make multiple appointments you must enable the checkbox for `Multiple Appointments` when creating a work order. Once this flag is enabled you can add additional appointments and modify existing ones.

> **Note:** Once the `Multiple Appointments` checkbox is enabled, it cannot be disabled.

With multiple appointments, work order capacity can be over allocated, because such jobs are usually scheduled for experienced professional. For example, if it requires 35 hours to complete a custom job, then a total of 5 working days consisting of 8 hours each day could be allocated for a specific resource.

10.5.4 Provided Service Work Order

Each provided service line on the work order corresponds to a single line on the order. Customers order provided services in the same way as they order products. Services are modeled within Selling and Fulfillment Foundation as service items. Services can be ordered alone or along with a product when purchased. Including the provided service line on the work order does NOT imply that all product lines associated with that provided service line are completed on that work order.

10.5.4.1 Provided Service Work Order with Products to be Delivered

When a customer buys a product and orders installation and delivery at the same time, the installer delivers the product. To allow for such cases you can make use of the `Products To Be Delivered` option in the `Work Order Details` screen. As a result, a single line that provides both the provided and delivery services is created.

Lines on a work order can be from multiple sales orders. For example, in the case of a product exchange, removal and uninstallation of the old product are able to be included on the same work order as delivery and installation of the new item. However, the delivery service lines are ignored for service items or skills when looking for available resource pools to accommodate the appointments.

> **Note:** If a work order consists of both provided and delivery service lines, then the work order is considered to be a provided service. The service item UOM is same as the provided service capacity UOM. The `ServiceItemGroupCode` is 'PS'.

10.5.4.2 Synchronization of Work and Exchange Orders

Lines on a work order can be from multiple sales orders. For example, if a customer wishes to return a newly installed dishwasher and replace it with one of a different color, a return order containing the uninstallation and pickup of the old dishwasher, along with the dishwasher itself is created. An exchange order is also created containing the new dishwasher, along with delivery and installation services for that new dishwasher. The user then has the option of generating a single work order that contains the following:

- New dishwasher
- Dishwasher delivery (Delivery Service)
- Dishwasher installation (Provided Service)
- Old dishwasher
- Dishwasher pickup (Provided Service)
- Dishwasher uninstallation (PS)

This enables the user to create only one appointment for both tasks.

10.5.5 Delivery Service Work Order

Work orders can contain delivery items with associated products. This is essential if the products to be delivered are large and the installer cannot deliver them or if the customer has ordered just delivery without installation. Including the delivery service line on the work order does

NOT imply that all product lines associated with that delivery line are delivered on that work order.

Work orders can also contain stand-alone delivery service lines, that can later be associated with a product line if needed.

Note: Only product lines included in the work order are delivered as part of the work order execution.

10.5.6 Dynamically Determining Work Order Type

The work order type, either Provided Service (PS) or Delivery Service (DS), is automatically determined based upon what types of lines are present on the work order. This determination can occur at order creation or during a change in the work order. Work order type is based on the following logic.

- A work order is considered to be a PS work order if there is at least one PS line present.
- A work order is considered to be a DS work order if all the lines present are DS lines.

The work order type can be switched dynamically when lines are added or removed from the work order. For example, on a PS work order that contains both PS and DS lines, removing all of the PS lines results in the work order becoming a DS work order.

Because updating the work order type changes the item group code, the following information is also be updated automatically:

- Service Item Group Code
- Required Capacity
- Capacity Unit of Measure
- Complexity level

The resource pool for the work order appointment is also updated upon a work order type change.

Note: If the Resource Pool was manually passed on the appointment, the appointment must be retaken manually after a work order type change.

10.6 Service Work Order Creation

A service work order is created from a sales order when there are service lines associated with the product. The creation of a work order does not change the status on the order and is allowed for order lines that have not yet been scheduled. Each work order relates to a single order line. However, you cannot include a work order for order lines that are completely shipped or cancelled. Keeping this consideration in mind, any number of provided service, delivery service, and product lines can be added to a work order.

Additionally, certain items can be configured as service tools that can be added to work orders. Typically, those are items that are used by service resources to perform the necessary tasks on the work order, for example a ladder, or a tool kit.

The following validations performed at work order creation:

- All provided service lines should have the same unit of measure.
- If the work order is a delivery service order without any provided service lines, all of the service lines should have the same unit of measure.
- The `Products To Be Delivered` option can only be set for provided service or delivery service work orders.
- All the lines that are included in the work order should have the same shipping (Ship To) address.

10.6.1 Provided Service or Delivery Service Work Order Determination

The type of work order is defined using the following criteria:

- If only delivery service lines and product lines are included in the work order, the work order is termed as a delivery service work order.

- If there is at least one provided service line present in the work order with the following combinations, then the work order is a provided service work order:
  - Provided service lines.
  - Provided service lines along with product lines.
  - Provided service lines along with product lines and delivery service lines.

When the work order contains delivery service lines only, you cannot add a provided service line. If the work order is a provided service work order, then you can add delivery lines but the work order type does not change.

10.6.2 Work Order Node Determination

Each work order should be created for the specific node which owns the work order execution. Selling and Fulfillment Foundation determines the work order node based on the following criteria:

- If a node is specified during work order creation it is used.
- If a node is specified on any of the service lines, that node is used as the work order node.
- If capacity is blocked by an order line, the node of the resource pool that was used to block the capacity is used.
- If there is no predefined node on the work order or the service lines, the sourcing rules for either provided service (for a provided service work order) or delivery service (for a delivery service work order) is used. In this case the primary node is used as the work order node.

> **Note:** In all of the above situations, if there is more than one node specified for the service lines, a work order node cannot be created. Therefore if you do not specify the work order node during work order creation, you have to make sure that you can cite only one node in all of the service lines.

A node cannot be changed on the work order. If you have to change the node, the work order must be cancelled and recreated.

10.6.3 Work Order Provider Organization Determination

Selling and Fulfillment Foundation determines the work order provider organization based on the following criteria:

- If the work order is a provided service work order, the owner of the work order ship node becomes the provider organization. The provider organization is ALWAYS determined when the work order is created. Therefore the provider organization cannot be changed on the work order after creation.
- If the work order is a delivery service work order, then the provider organization can be specified during work order creation. However, no validations are made to ensure that the provider organization has resource pools defined for the work order ship node.

Even after meeting this criteria, if the provider organization cannot be determined, it is left blank on the work order. It can be either set later by passing the Provider Organization using the `modifyWorkOrder` API or by making the first appointment. In the latter case, the provider organization for the resource pool is assumed for the work order.

10.6.4 Capacity Calculations for Work Order

Each service line added to a work order, except when provided and delivery service lines are mixed on the same work order, contributes to the computed capacity. The capacity for an item can be computed either using product association or the ordered quantity.

For example, if the provided service line has 2 associated product lines with a quantity of 3 in each line. Then the total required capacity for the work order is 2 x 3 x CapacityMultiplier. The `CapacityMultiplier` is a multiplication factor which varies between a provided or a delivery service work order.

Capacity Calculation for a Delivery Service Work Order

ComputedCapacity = Max (fixed capacity of each delivery service line) +

Sum (variable capacity of all delivery service lines)

The product catalog is used to determine the fixed and variable capacities. If capacity is estimated based on the ordered quantity, then computed capacity is calculated for a fixed capacity.

Additionally, required quantity of a delivery service line is computed based on the product lines included in the work order. However, when the service lines are added or removed, the requested quantity is recomputed unless the capacity overridden option is enabled in the work order. When a product that must be delivered, is not associated to a delivery service line on the sales order, the association is computed from the product catalog for variable capacity.

If there is more than one delivery service line on a delivery service work order, a variable capacity is computed for the item that is associated with the product catalog. In the case of more than one delivery service lines, the capacity for the delivery service work order is chosen at random. When a product is not associated with any delivery service item in the catalog it contributes `0` capacity.

Additional factors can also be taken into account when calculating required capacity on a delivery service work order. For more information about additional fixed capacity and supplemental capacity, see Section 9.3.5, "Resource Pools". For more information about capacity impact for answers, see the *Sterling Distributed Order Management: Configuration Guide*.

Capacity Calculation for Provided Service Work Order

ComputedCapacity = Sum (fixed capacity on all provided service lines) +

Sum (variable capacity of all provided service lines)

The capacity is considered to be fixed, when the computation is based on the ordered quantity.

RequestedCapacity = ComputedCapacity + AdditionalRequestedCapacity

The `AdditionalRequestedCapacity` can be used to manually specify any additional capacity required. This value can be a positive or negative number. For example, when the installer delivers the product and the field supervisor knows that it requires an extra 30 minutes to pick up the product from the hub. In this case, the additional 30 minutes is considered as `AdditionalRequestedCapacity`.

However, this additional capacity can be overridden on the work order creation by manually specifying the requested quantity. The use-case of this situation arises for complex jobs where required capacity cannot be derived based on the service item associations.

Once requested quantity is overridden any changes to the quantities on the order lines do not result a change in the requested capacities. However, when the last service line is cancelled then the entire work order is cancelled.

The `QuantityRequested` indicates the measure of capacity needed to complete the work order. The blocked capacity is provided as `AllocatedQuantity` while planning appointments. In most cases, the allocated and requested quantity remains the same once an appointment creation process is complete.

Additional factors can also be taken into account when calculating required capacity on a provided service work order. For more information about capacity impact for answers, see the *Sterling Distributed Order Management: Configuration Guide*.

10.6.5 Product Reservation for a Service Work Order

Product reservation is attempted by the schedule transaction which is called to reserve an item based on the requested ship date specified on the sales order. If the product cannot be scheduled for release, the schedule transaction puts the product line in reserved status.

Once the product line is reserved, the ship node on the work order line is populated as a non-firm ship node. When determining the ship node for product lines on the work order, Selling and Fulfillment Foundation considers the following:

- If product lines have predefined ship nodes, they are used for product reservation.
- If no node was specified on the product line and it is a delivery service work order, the ship node on the delivery service line is used.
- If no node was specified on the product line and it is a provided service work order then the following considerations are made:
  - The node of the service installer is used as the delivering node of the product. If this option is selected, the ship node on the work order line is used to attempt product reservation. This option can be used when inventory is physically located at the installer hub.
  - The standard rules for determining the delivering node of the product is used. This option can be used when it is insignificant if

the service installer delivers the product or the product is separately delivered.

10.7 Confirming a Work Order

To confirm a work order, you need to specify the appointment that is being confirmed, the outcome of the appointment, and the execution details. However, the delivery service lines on the work order are not marked as completed. They are moved to delivered status. The work order is considered complete when all the provided service lines are completed and all products are delivered. In a single work order confirmation, multiple appointments can be completed. If the `confirmWorkOrder` API is called to confirm the work order then all open appointments are cancelled and allocated capacity are removed.

Selling and Fulfillment Foundation also provides the capability to confirm work orders without specifying the appointment details. If, as part of the work order completion, all service lines (provided service and delivery service) are completed and all products are delivered, open appointments on the work order are resolved as follows:

- For a work order with a single appointment, the appointment is marked as completed.
- For a work order with multiple appointments, all appointments in the past are marked as completed, and future appointments are cancelled.

10.8 Work Order Cancellation

Work order cancellation depends on the status of the service lines. If any of the service lines were completed or any of the product lines were delivered, then the work order is marked as completed. Only appointments that are open are cancelled and the associated capacity reservation is removed.

It is also possible to request cancellation of a delivery, after a product has been shipped. If a customer decides that they no longer want the item they originally ordered, but the item has already shipped, a stop delivery request can be created. If this request is successful, a return order is automatically created.

10.9 Appointment Status

An appointment status depicts the status of the work order appointments. The appointment status is associated with the blocked capacity. The appointment start time represents the minimum of all appointment start times that are currently blocking the capacities. The appointment end time represents the maximum of all appointment end times that are currently blocking capacities. Whenever appointments are made without a capacity check they can be overridden.

The possible statuses for appointments are:

- Open
- Completed
- Failed
- Cancelled

Open

When an appointment is taken against a work order the appointment status is set to Open. This is true even if the capacity is overridden or resource pool requires confirmation. This status indicates that the appointment has been taken.

Completed

When an appointment is recorded as a successful execution the appointment status is changed to Complete. Once the appointment is marked as complete the appointment details cannot be changed.

Failed

If the customer is not available the appointment is changed to Failed. The appointment cannot be changed to any other status once it is marked as Failed. If the customer is available at a later date, the appointment should be re-created.

Cancelled

The appointment status is set to Cancelled if an appointment is cancelled as a result of work order cancellation or work order completion. When a work order is being confirmed and if all the services are

completed along with the product delivery, all appointments that are not failed or completed are marked as Cancelled.

10.9.1 Procedure involved in taking Appointments

You can make appointments directly on the work order once it is created. Appointments are made for a resource pool or a service resource. Only the resource pool that provides all the skills or service items for all the service lines that are required, are included on the work order. To increase the option of selecting more than one resource pool you can opt to ignore the product availability while checking for resource pool availability. Appointments can be slot-based or calendar-based.

Slot-Based Appointments:

When an appointment is slot-based, the start and end time of the appointment indicate the actual time slot. If the resource pool is time-based, the difference between promised appointment start and end dates indicate the required capacity. Slot-based appointments can be made for either a resource pool or a service resource.

Calendar-Based Appointments:

When a calendar-based appointment is made for a non time-based resource pool, the required capacity should be specified. If a service resource is specified together with the resource pool, capacity, service region and skill validations are performed.The required capacity is blocked against the specified service resource or resource pool.

Work Order Creation and Appointment Planning Example:

The following example demonstrates the creation of a simple work order and appointment planning:

A customer orders a TV with installation that requires a total of 2 hours, 0.5 hour is fixed capacity and 1.5 hours variable capacity.

The customer also orders a DVD player with installation that requires 1 hour of fixed capacity and 0.5 hour of variable capacity. The sales person takes the customer's order, and creates a work order by selecting both installation service lines (TV-Install and DVD-Install) and specifies the ship node as his own store. TV-Install and DVD-Install have the current work order field stamped with WorkOrderKey.

Work Order is created with provider organization as the owner of the Ship Node. The requested capacity is 3 hours (sum of required quantities 2 and 1 hours). The sales person checks for an appointment and offers available slots to the customer. The slot on November 12 from 9:00 AM to 12:00 PM is selected. The appointment status shown as Open, and the appointment is shown on the work order header and both order lines (TV-Install and DVD-Install). Thus 3 hours of capacity is blocked between 9:00 AM - 12:00 PM on November 30 for specified resource pool which in this case is the retail store.

10.9.1.1 Pre-Calling before the Appointment Date

When appointments are made for distant dates, it is preferred to make a pre-call closer to the service date as a reminder to the customer. Selling and Fulfillment Foundation supports a sophisticated process for managing and monitoring pre-calls which are not mandatory.

Selling and Fulfillment Foundation supports confirming a appointment even when a pre-call attempt fails.

When the service provider attempts to pre-call a customer, the pre-call could result in one of the following:

- **Successful Pre-Call** — Occurs when a customer is reached during a pre-call attempt, and the customer confirms that the previously taken appointment is fine.
- **Failed Pre-call** — Occurs when a customer could not be reached during a pre-call attempt. However, an attempt may be made later to re-confirm the appointment with the customer. Selling and Fulfillment Foundation supports confirming an appointment even if the pre-call attempt is unsuccessful.
- **Change of Appointment** — During a pre-call attempt, a customer may requests for a change in the appointment. Based on how distant in the future the new appointment is, another pre-call attempt may be made at a later point in time that is closer to the new appointment.

10.10 Invoicing Provided Service and Delivery Services

Provided and stand-alone delivery service invoicing is done based on the entire ordered quantity on the sales order line.

Delivery service with associated product lines are invoiced when the associated products are invoiced. If the ordered quantity is the same as the quantity specified when creating the delivery service line, then the complete quantity is invoiced at the same time as the product lines.

For example, consider the following two cases:

Case 1: Entered Quantity Strategy

If the entered quantity strategy is used to invoice delivery service lines with associated product lines, the entire delivery service line quantity is invoiced when the product lines are invoiced.

Consider that the product quantity is 10 and the quantity entered on the delivery service line is 5. The delivery service line is invoiced for its entire quantity when product quantity is invoiced, regardless of whether the product quantity is partially or completely invoiced.

Case 2: Associated Quantity Strategy

If the associated quantity strategy is used to invoice delivery service lines with associated product lines, the invoiced quantity on the delivery service lines is computed based on how much product quantity is invoiced.

Consider that the product quantity is 10 and the delivery service line quantity is 5. The delivery service line is invoiced for 1 when the first 2 product units are invoiced. In this case, the product quantity and the delivery service line quantity maintain a ratio of 2:1 respectively.

10.11 Work Order Pipeline

When modeling your business process, you can create pipelines for work orders. A pipeline consists of the different statuses a document goes through during its life cycle. You can also set up transactions as they pertain to the pipeline that you are configuring.

The Work Order document uses the value-added service pipeline. Work orders can also be part of the scheduling activities in the order pipeline.

10.11.1 Value-Added Services Pipeline for Work Orders

From its creation, a work order flows through a set of transactions and statuses until its completion. This chain of transactions and work order statuses is called the Value-Added Services (VAS) pipeline. You can configure the VAS Pipeline to meet any special business requirements that you have. However, every VAS generally begins with the transaction that creates the work order, and ends with a transaction when the work order is completed.

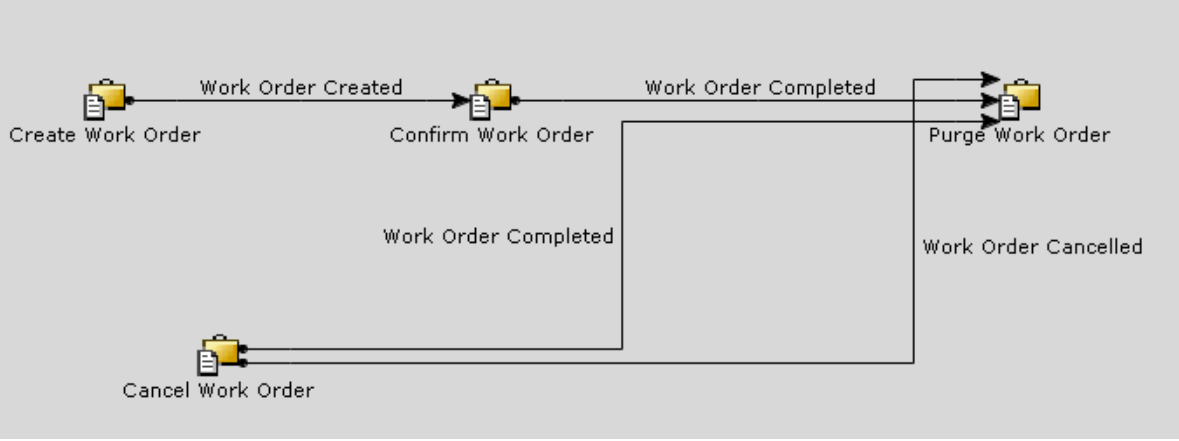
A transaction is an occurrence that needs to be tracked and can call certain actions to be performed. An order status describes what state work order is in.

The following statuses may be used in a VAS pipeline, depending how it is configured within your system:

- Created - the work order is created
- Confirmed - the services in the work order are completed
- Cancelled - the request for services in the work order is cancelled

Figure 10–2, "VAS Pipeline" shows the default VAS pipeline for Selling and Fulfillment Foundation.

Figure 10–2 VAS Pipeline



10.12 Compliance Services

Compliance services supply the ability to customize items for a buyer that meet that buyer's detailed specifications. For example, a vendor may supply a refrigerator that can be customized with a department store chain's brand, *Big Store*. When the refrigerator is shipped, it has a small emblem that identifies it being the department store brand and all the supporting materials, such as warranties and mail-in rebates, are specific to the department store, rather than the original vendor.

To support buyer compliance, the `Requires_VAS_Compliance` flag must be enabled in the Buyer Organization configuration. This indicates that the buyer may have orders for items that are made to its specifications.

When configuring information about the buyer, you can identify item classifications that should have a compliance service performed upon it. This compliance service defines activities that should be performed to prepare the items in that item classification to meet the buyer's requirements. Items which require compliance services to be performed can not be kits.

For example, there may be a classification of "Music CDs". For all music CDs that this buyer purchases, a sticker advertising a new music service is applied to the CD case. When configuring the buyer to support this scenario, a compliance service is created to describe how to apply the sticker. This compliance service is then associated with the Item

Classification "Music CDs" in the buyer's compliance services configuration.

10.12.1 Configuring Compliance Services

When configuring a compliance service, you can indicate whether work orders should be generated when the compliance service is performed. Under some circumstances, the task to be performed may not effect the inventory scheduling or the general processing, so a specific work order is not needed.

For example, a buyer wants a mail-in-rebate coupon added to all CD players. In this case, the compliance service that describes this process may be configured so that a specific work order is not generated because there is no change to the overall inventory scheduling.

In a different scenario, a shirt may be embroidered with a football team logo. This affects available inventory and should be configured to generate a work order, both to provide tracking for the processing, and to enable the use of the run number optimization of the compliance service processing. When the work order is generated for this service it is not reflected on the sales order. Also, if the work order is cancelled, the change is not reflected back on the sales order. In cases when an order is released and an inventory check is required, the sales order is automatically back ordered since the work order has been cancelled and no inventory is available. If an inventory check is not required in this case, the sales order requires a manual change to back order.

10.12.2 Using Run Quantity to Optimize Compliance Processing

When a compliance service is configured, you can configure run quantity to control your processing of special items. A *run quantity* is used to calculate how many of the item should be created when there is insufficient inventory to meet an order for the item.

When an order for an item that requires compliance services is placed, the system tries to schedule against available inventory that has been set aside to fulfill these special orders (or segmented) at the ship node specified on the order. If inventory is not available at the ship node, the system tries to schedule against any segmented inventory at configured procurement nodes. If there is still not enough inventory to satisfy the

order, a work order is created, based on the compliance services for the item. This work order is used to manage the specific requirements for supplying the item.

For example, *Big Store* purchases over 150 refrigerators each week. The compliance service for those refrigerators is configured so 100 refrigerators should be made when a work order is created.

Another buyer, *Local Store,* purchases 10 to 15 refrigerators a week. When configuring a compliance service for them, only 10 refrigerators should be created when a new work order is created. The Table 10–1, "Compliance Services and Inventory" summarizes the weekly purchases and the configuration for the compliance service.

Table 10–1 Compliance Services and Inventory

Buyer	Refrigerators purchased weekly	Compliance Service: Run Quantity
Big Store	150	100
Local Store	10-15	10

The current inventory for the customized refrigerators is described in the Table 10–2, "Current Inventory for Custom Refrigerators".

Table 10–2 Current Inventory for Custom Refrigerators

Buyer	On Hand
Big Store	21
Local Store	7

If *Big Store* orders another 30 refrigerators, there is not enough inventory available to supply them. A work order is created which specifies the steps for the compliance service. The run quantity for the compliance service is 100, so a total of 100 refrigerators are customized to the Big Store's requirements.

If *Big Store* is having a sale and orders 150 refrigerators, a work order is created for 200 refrigerators (two runs of 100 each). This occurs because only 21 refrigerators are onhand and when added to a run quantity of

100, only 121 refrigerators would be available. Therefore the run quantity is used repeatedly to create enough to meet the requirements of the order.

> **Note:** When the ordered quantity is not equal to the run quantity, two inventory checks occur during scheduling instead of one.

If the Local Store orders another 7 refrigerators, no work order is created, and no additional refrigerators are created. This completely depletes the stock of Local Store refrigerator. If the Local Store orders 7 or more refrigerators, then an additional 10 refrigerators are created.

11

Supply Collaboration

Supply Collaboration coordinates and collaborates direct material supply plans, purchase orders, replenishment, inventory and inbound fulfillment across multiple business units, division, suppliers, outsourced manufacturers and transportation providers.

Supply Collaboration involves managing the purchase order process for your products.

11.1 Purchase Orders

In Selling and Fulfillment Foundation, a purchase order can be broken down into three levels, the purchase order header level, purchase order line level, and purchase order release level. The purchase order header level contains all of the purchase order lines in a purchase order, the purchase order line level is broken down by each individual purchase order line, and the purchase order release level contains all of the lines that have been released to a ship node.

A purchase order is made up of purchase order lines. A purchase order line is identified by a unique item, unit of measure, and product class.

11.2 The Purchase Order Execution Pipeline

Beginning from its creation, a purchase order flows through a set of transactions and statuses until its completion. This chain of transactions and purchase order statuses is called the purchase order execution pipeline. There are two default pipelines the standard Purchase Order Execution pipeline for organizations that own their nodes and the Drop Ship Purchase Order Execution pipeline for organizations that use drop ship nodes for purchase order handling.

A purchase order execution pipeline provides Selling and Fulfillment Foundation with a means to perform actions, such as sending notifications or logging alerts, dependent on the location of the purchase order in the pipeline. It also provides you with a means to track a purchase order from creation to completion and perform any manual interventions, if necessary.

The purchase order execution pipeline from which your business runs is unique to how your system administrator has set up your business environment. However, every purchase order execution pipeline generally begins with a transaction that creates a purchase order and ends with a transaction that indicates a purchase order has been shipped.

A transaction is an occurrence that needs to be tracked and can call certain actions to be performed. A purchase order status describes what state a purchase order is in and moves it from transaction to transaction.

The following statuses may be used in a purchase order execution pipeline, depending how it is configured within your system:

- Accepted - The negotiated terms have been accepted by both parties and the purchase order is ready to be released.
- Awaiting Chained Order Creation - A chained order must be created and sent to the applicable node.
- Backordered - The purchase order has been created, but there is not enough inventory to schedule it. The purchase order remains backordered until inventory is available.
- Backordered From Node - The purchase order has been created and released to the node, but the node does not have enough inventory to fulfill it.
- Being Negotiated - The purchase order is being negotiated in the negotiation pipeline.
- Cancelled - The purchase order has been canceled.
- Chained Order Created - A chained order has been created from the purchase order.
- Created - The purchase order has been created.
- Draft Order Created - A draft purchase order has been created in the Create Purchase Order Console. All aspects of this order can be modified until it is confirmed.

- Included In Shipment - The purchase order has been added to a shipment for delivery.
- Receipt Closed - The necessary return handling for the purchase order has been completed and the return is closed.
- Received - The purchase order has been received by the Buyer.
- Received As Components - If an item in the purchase order consists of kit components, this status indicates that the Buyer has received all components.
- Released - There is enough inventory to schedule to the purchase order for fulfillment. The purchase order is released to the Application Console, the Selling and Fulfillment Foundation Warehouse Management System, or another third-party warehouse management system.
- Reserved - The purchase order has been created, but it is not ready to schedule yet.
- Scheduled - The applicable node(s) have the inventory to fulfill the purchase order and can be scheduled for release.
- Sent to Node - The purchase order is sent in the form of an order release.
- Shipped - The purchase order has been shipped.
- Shipment Delayed - The all or part of the purchase order shipment has been delayed.
- Shorted - The purchase order contains less quantity than requested. The order is closed.
- Unreceived - The purchase order has not been received by the buyer.
- Unscheduled - The purchase order has been removed from Scheduled status and any inventory that has been reserved for the purchase order at the scheduled node(s) is canceled.

Figure 11–1 Purchase Order Execution Pipeline

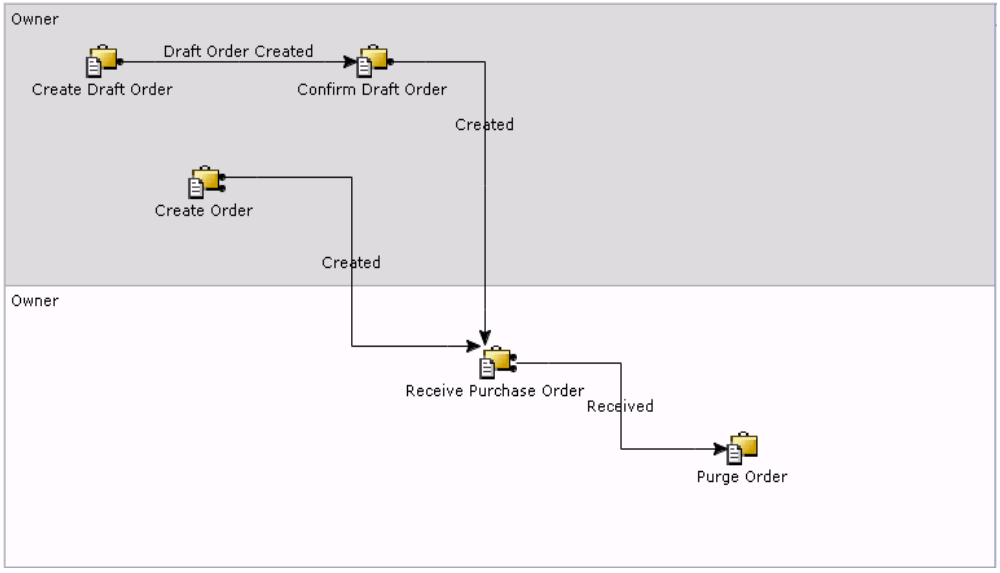
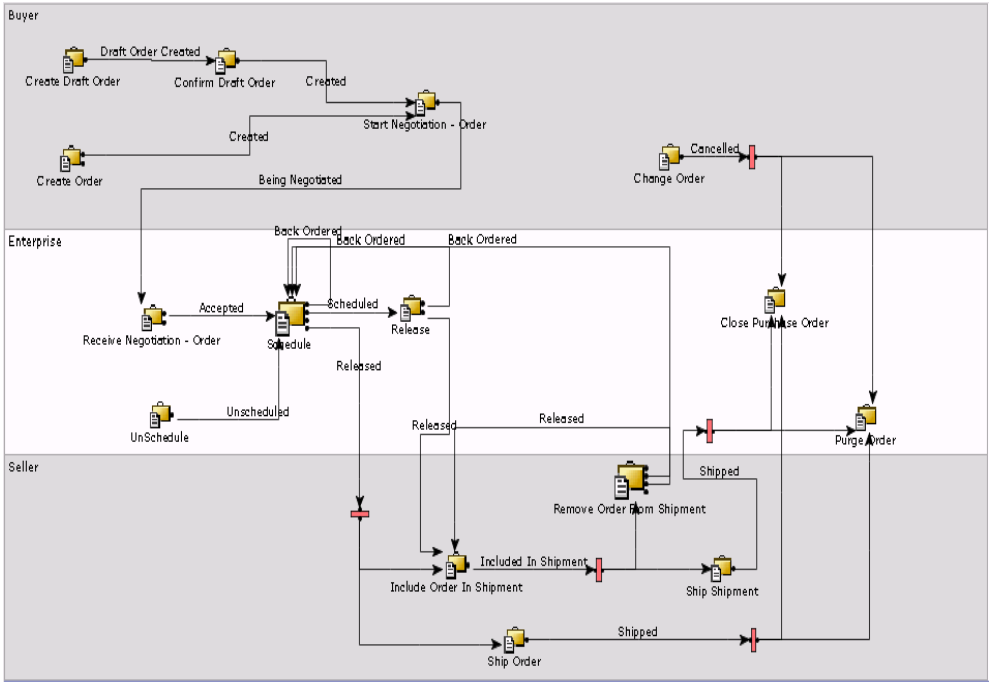


Figure 11–2 Drop Ship Purchase Order Execution Pipeline



Note: The default pipeline and statuses can be configured as per your business practices. For more information about configuring purchase order execution pipelines and statuses, see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide*.

12

Payment Systems

Selling and Fulfillment Foundation can be used to carry out the critical payment-related processes during order management processing and enables you to integrate with external payment processing systems.

Payment processing involves payment authorization, settlement, and invoicing. This chapter describes the payment processing features, modules, components, and events, related to authorization and settlement that occur at various times in the Selling and Fulfillment Foundation order lifecycle.

12.1 Payment Processing Overview

Selling and Fulfillment Foundation provides several different payment processing features:

- Multiple Payment Methods
- Synchronous Payment Processing
- Asynchronous Payment Processing
- External Collection through Accounts Receivable
- User Exits and APIs Provided to Handle Custom Logic
- Charge Consolidation
- Pre-Payments Recorded
- Request Amount Returned
- Delayed Reauthorization
- Charge Transaction Request Identifiers

- Reverse Authorization
- Pre-Settlement Support
- Invoicing
- Returns Payment Processing and Refund Fulfillment
- Exchange Payment Processing
- Draft Order Payment Processing
- Manual Interventions
- Multiple Control Levels
- Payment Rules

12.1.1 Multiple Payment Methods

Selling and Fulfillment Foundation enables you to associate an unlimited number of payment methods with an order. Furthermore, within a single payment type (such as credit cards), Selling and Fulfillment Foundation permits an unlimited number of individual credit cards. For example, a specific order can be paid for using a discount coupon, a gift check, and two credit cards.

12.1.2 Synchronous Payment Processing

The Selling and Fulfillment Foundation transactions and user exits can be configured to contact an external payment processing system synchronously and wait for the result of the authorization or charge request. Depending on the success or failure of the request, the payment status of the order is updated immediately.

12.1.3 Asynchronous Payment Processing

The Selling and Fulfillment Foundation transactions and user exits can be configured to communicate asynchronously with external payment processing systems. The requests for authorization and charge then can be dispatched in a batch mode, and updates can be made all at once, based on the results of the request. The payment status of the order in Selling and Fulfillment Foundation shows whether the request is still outstanding or has been processed.

12.1.4 External Collection through Accounts Receivable

Selling and Fulfillment Foundation permits integration with external systems that handle accounts receivable information. If this is enabled, all the invoice details are published to the external accounts receivable system. The settlement process is not handled by Selling and Fulfillment Foundation. Any collections occurring outside Selling and Fulfillment Foundation are not reported back.

12.1.5 User Exits and APIs Provided to Handle Custom Logic

User exits get invoked selectively, depending on payment type. User exits can be customized to suit your needs. APIs have also been provided to enable customized payment processing.

12.1.6 Charge Consolidation

Charge consolidation reduces the number of synchronous calls made to the external payment processing system. You set up a charge consolidation timeframe window, based on each payment type defined for a seller. Later in production mode, when similar requests for a particular order arrive within the specified timeframe, they are grouped together and then the external call is made.

12.1.7 Pre-Payments Recorded

Payment authorizations and collections can be made on the front end and then communicated to Selling and Fulfillment Foundation when the order gets created. Selling and Fulfillment Foundation can then record these pre-authorized and pre-collected amounts. If these are only partial payments, the amount is deducted from the total order amount and then the typical payment processing behavior continues for the remaining order amount.

12.1.8 Request Amount Returned

The getOrderDetails API returns the request amount to be authorized on orders, including draft orders. The request amount equals the following subtracted from the total of the order:

- MaxChargeLimits of active payment methods

- Charges made with currently suspended payment methods
- The sum of any credits not tied to payment methods

Payment methods marked UnlimitedCharges = Y are counted as having a limit of 0. If there are extra charges or authorizations on the order, the request amount may be negative. For example, if an order has a total of $100, and the following payments, the request amount is $60 ($100-0-$30-$10):

- Gift card with a maximum charge limit of $30
- Suspended credit card with a maximum charge limit of $20 ($10 charged)
- Credit card with unlimited charges

12.1.9 Delayed Reauthorization

The delayed reauthorization feature controls the number of payment reauthorizations that may occur when scheduling orders against backordered or pre-ordered inventory. While waiting for the ship date to approach, payment authorizations can expire and reauthorize several times, locking up customers' credit lines. This also costs money when interfacing with a payment gateway. Two delayed reauthorization configuration options limit the number of reauthorizations to configurable points in the order cycle.

Note: Delayed authorization is not supported for draft orders.

12.1.10 Charge Transaction Request Identifiers

The charge transaction request identifiers feature provides a mechanism to control when payment authorizations occur on an order and the amount of the authorization. Instead of authorizations being processed automatically at the order level, this feature provides a finer granularity of control for authorizing payments pertaining to orders. Authorizations are tied to identifiers, which represent an entity or group of entities in an order. For example, payment authorizations can be processed at the

order release level or order line level, instead of only at the order level, by creating request identifiers for either of the entities.

Note: The charge transaction request identifiers feature does not support the Settlement Required payment option.

12.1.11 Reverse Authorization

Some credit card companies charge additional fees to merchants who do not explicitly reverse an unused credit card authorization. The Reverse Authorization feature enables merchants to implement an authorization strategy to reverse unused authorizations before they expire. This feature also provides a way to handle differing authorization and settlement amounts through user exits.

12.1.12 Pre-Settlement Support

Pre-settlement enables partial or complete charging of an order at any stage of order processing. This is typically used in pre-ordering or store pick-up situations where the item has already been sold to the customer and has been set aside. Therefore, the collection process must be initiated before the actual shipment of the product.

12.1.13 Named User Exits

You can invoke a User Exit directly by implementing the new API called invokeUE. This is useful in situations where you need to invoke certain functions such as Fraud Check that are implemented as User Exits. The invokeUE API is the entry point to any application that wants to invoke a named User Exit. The invokeUE API internally calls the User Exit as specified in the input XML and will pass back the output of the User Exit to the caller. For more information about invokeUE API, see the *Selling and Fulfillment Foundation: Javadocs*.

12.1.14 Draft Order Payment Processing

Payment processing can be enabled for draft orders by setting the Enable Payment Processing for Draft flag at the hub level. This allows processing of payments on draft orders before the draft orders are confirmed. Additionally, the charge transaction table and the payment status are

populated when draft orders are created. This flag is on by default. Draft order payment processing includes the following features:

- The processOrderPayment API accepts and persists new payment methods
- Displays error messages for failed authorizations
- Modifies failed payment methods
- Ignores charges on draft orders
- Perform payment processing based on charge sequences
- Consolidates charge transactions

The processOrderPayment API Accepts and Persists New Payment Methods

The processOrderPayment API is enhanced to accept and persist new payment methods on orders. This allows uncharged and unauthorized payment methods to be modified and deleted on the order.

Displays Error Messages for Failed Authorizations

Error messages are returned in the processOrderPayments API output XML and stored in the YFS_PMNT_TRANS_ERROR table. To retrieve error messages after payment processing is completed, invoke the getOrderDetails API.

Modifies Failed Payment Methods

If a payment method on a draft order fails to authorize, the payment method can be deleted or modified. In this case, the order payment status must be Failed and the charge transaction status must be in error. To modify the payment method, change any of the attributes of the payment method, except payment key and payment type.

Ignores Charges on Draft Orders

When enabled, the Ignore Charge On Draft flag is used to ignore charge requests when calculating the request amount on a draft order. For example, if a draft order totaling $100 has a credit card authorization of $20 and check (charge request) for $20, the getOrderDetails API returns a request amount of $80. The Ignore Charge On Draft flag is turned off by default and configured at the hub level. The Enable Draft Order

Processing flag must be set to On before you configure the Ignore Charges On Draft flag.

Performs Payment Processing Based on Charge Sequences

In the processOrderPayment API, the charge sequence for each payment method determines the order in which authorizations and charges are processed on draft orders. By default, payment methods have equal charge sequences, and are thus processed in random order.

Consolidate Charge Transactions

A charge transaction record is created each time an order is modified. For this reason, many charge transaction records may exist for an order. The confirmDraftOrder API consolidates all charge transaction records for an order into a single record.

12.1.15 Invoicing

The invoicing process typically initiates the settlement process in the order life cycle. Invoicing can be done either through time-triggered transactions or APIs.

Shipment Invoicing occurs after the order has shipped. It initiates the collections process.

Order Invoicing enables the settlement process to start at any stage of the lifecycle of an order, not just after shipment. This can be used in situations where there is no shipment or instances when the invoice needs to span multiple shipments.

The Provided Service and Delivery Service is invoiced based on the ordered product line quantity. For more information, see Section 10.10, "Invoicing Provided Service and Delivery Services".

Return invoicing enables refunds to be issued to the customer during the return process. The refund can be configured to occur either before or after the receipt of the returned goods. If the return was created by a gift recipient, Selling and Fulfillment Foundation does not issue the refund to the buyer on the sales order, but to the gift recipient instead.

Any modification to order taxes and changes during invoicing time will update the order only when the order is completely invoiced. In addition, any modifications to line taxes and changes on the line during invoicing

time will update these order lines only when the order line is completely invoiced

Invoices can be published immediately after creation (generally for external collections) or after settlement within Selling and Fulfillment Foundation.

Pro Forma Invoicing

In many business scenarios, if there are multiple shipments for each order, charges and taxes are split up between the shipments. Depending on implemented business processes, these amounts may be split amongst various shipments in several ways. To appropriately charge and tax subsequent shipments of an order, previous charges and taxes must exist in the database to be persisted against.

A Pro Forma invoice, an invoice generated upon shipment creation, acts as draft invoice that persists charges and taxes in the database so that when charges and taxes are calculated for additional shipments, previous charges and taxes are taken into consideration.

For example, an order with two lines is created for $70. The enterprise uses incremental charges for shipping as follows:

- A $4.99 shipping charge is added to orders between $0 and $49.99
- A $6.99 shipping charge is added to orders between $50 and $99.99

Line1 on the order is $40, while Line2 on the order is $30. Without pro forma invoicing the following charges would occur upon invoicing:

Table 12–1 Example: Order Without Pro Forma Invoicing

Line	Amount	Charge
Line1	$40	$4.99
Line2	$30	$6.99

With pro forma invoicing, when calculating the shipping charge for Line2, the existing charge on Line1 is taken into consideration:

Table 12–2 Example: Order With Pro Forma Invoicing

Line	Amount	Charge
Line1	$40	$4.99
Line2	$30	$6.99 - $4.99 = $2.00

Cancelling Part of an Order with Pre-Collected Payments

If an order has not yet been shipped, but the payment for that order has already been pre-collected, cancelling all or part of the order could require a certain amount to be refunded. If that is the case, Selling and Fulfillment Foundation sends a notification that can be configured to be picked up by external payment systems with the amount refunded, if applicable.

For example, if an order for $140 has $100 worth of pre-collected payments and $50 are cancelled, an amount of $10 needs to be refunded to the pre-collected payment type. In that case, a notification is raised by Selling and Fulfillment Foundation for the $10 refunded.

Additionally, the `Deferred Credit On Return Required` rule can be enabled in the Applications Manager at the payment rule level. If that rule is enabled, and if a cancelled amount does not need to be refunded, Selling and Fulfillment Foundation sends a notification that can be configured to be picked up by external systems with that amount.

If, in the above example, the `Deferred Credit On Return Required` rule is enabled, another notification is raised by Selling and Fulfillment Foundation for the $40 that was cancelled but not refunded.

Price Changes Post Invoicing

Selling and Fulfillment Foundation allows for any charge or unit price change on an order to trigger the creation of adjustment invoices when the order has already been invoiced. This can be done by enabling the `Apply Changes To Invoiced Quantity` rule at the document type level for a given enterprise. The adjustment invoices are picked up by payment agents, and processed accordingly.

For example, some stores offer their customers a lowest-price guarantee. If a customer buys a pair of shoes for $50 and finds those same shoes at another store for $45, then should request for a $5 refund. If the `Apply Changes To Invoiced Quantity` rule is enabled and the unit price of the shoes is changed to $45 on the sales order post invoice creation, an adjustment invoice is created for $5 and a refund is issued to the customer for that amount.

Tax Changes Post Invoicing

Selling and Fulfillment Foundation allows for any change to the tax percentage on an order to trigger the creation of adjustment invoices when the order has already been invoiced. This can be done by turning on the `Apply Changes To Invoiced Quantity` rule at the document type level for a given enterprise. The adjustment invoices are picked up by payment agents, and processed accordingly.

For example, if a customer with tax exempt status makes a purchase but neglects to inform the store of his status until after the order has been shipped and invoiced, an adjustment invoice is created and a refund is issued to the customer.

When tax is applied on an order or an order line by passing tax percentage, the tax amount on the order, the order line, and the tax record are updated after invoicing. To reflect the correct tax amount on the order, the order line, and the tax record in the output of order APIs before invoicing, the correct tax amount for the corresponding tax percentage must be passed in the input or the recalculate tax user exits must be implemented.

12.1.16 Returns Payment Processing and Refund Fulfillment

During the return process, the charges that have been collected from the consumer must be credited back to them. The refunded amount is credited back using the original payment method.

12.1.16.1 Refund Sequence

If a sales order had used several different payment methods, the charges are refunded according to the refund sequence as defined in the Applications Manager. In the case of two or more payment methods having the same priority, the refund occurs in reverse order of the

payment sequence used in the sales order for those methods. This default sequence can be overridden by implementing the `confirmRefundPayments` user exit.

12.1.16.2 Pre-Payments

When the return is derived from a sales order that contains some pre-payments and some regular payments, Selling and Fulfillment Foundation first refunds against its own payment methods. If by that point the full amount has not been refunded, it uses the pre-payment methods.

12.1.16.3 Default Refund Payment Type

A default refund payment type can be defined so that if an order does not have any valid payment methods for issuing a refund to, the refund is issued to a new payment method of the default refund payment type. The default refund payment type can be overridden by implementing the `confirmRefundPayments` user exit.

12.1.16.4 Refunding the Same Account

The 'Refund to Same Account' `flag`, as defined in the payment type level, indicates whether or not the payment should be refunded to the same account that it was charged to, or create a new account. For example, if a customer placed an order using a Stored Value Card and the flag is enabled, then the refund will be made against that same Stored Value Card.

12.1.16.5 Refunding Against a New Payment Type

If the 'Refund to Same Account' flag is disabled, Selling and Fulfillment Foundation tries to refund against a new payment type as defined in the Applications Manager. If you select the same payment type as the new payment type, Selling and Fulfillment Foundation refunds to the same payment type, but under a new account. For example, if a payment is made on a Stored Value Card when the 'Refund to Same Account' flag is set to 'N' and the new payment type is set to Stored Value Card, then the refund will be issued against a new Stored Value Card.

If you do not want to create a charge request in YFS_CHARGE_ TRANSACTION table, then a refund to a new payment type is created. In this case, the incomplete charge transaction is raised and human intervention is required to complete the charge details for the payment

type. Once the details are entered, the request collection agent picks up the charges that should be refunded and move it to the PAID status.

12.1.16.6 Alternate Refund Payment Type with Constraint

There are cases where a constraint should be placed on refunding against certain payment types. For example, if a Stored Value Card costs the Enterprise $4 in production cost, and a refund amount is less than $5, the Enterprise may rather issue a check. To achieve this, a payment type can be configured to have an alternate refund payment type, which refunds are issued on when the refund amount satisfies the constraint.

12.1.16.7 Refund Fulfillment Order

A refund fulfillment order is created by the Payment Collection transaction when the 'Create Refund Fulfillment Order Using' flag is enabled. When this flag is enabled, a refund fulfillment order is created instead of a new payment method. For example, a customer placed an order using a gift card and then returned the order. Typically, the customer has a new gift card shipped to him in this case. A new fulfillment order is automatically created to ship the gift card.

Note: Charge Instead of Authorize payment types will be refunded, even if settlement is not required.

12.1.17 Exchange Payment Processing

When an exchange is created, the total order amount of the return order is added to the PENDING_TRANSFER_IN field of the exchange order. This value is always kept synchronized on the exchange and the return order. For example, if an exchange is initially derived from a return order with a total amount of $40, and a line is later on added to the return that increases that amount to $60, the PENDING_TRANSFER_IN of the exchange is increased to $60 as well.

For advanced and regular exchange orders, if the total amount on the exchange order exceeds the amount in PENDING_TRANSFER_IN, additional payment information needs to be entered.

A TRANSFER_IN charge record is created on the exchange order while a TRANSFER_OUT charge record is created on the order the amounts have been transferred from.

In the case of a pre-paid exchange, payment information has to be collected up front for the entire amount of the exchange.

12.1.18 Manual Interventions

If required, users can override standard behavior, using a variety of UI controls, APIs, transactions, and user exits. Access to these manual activities can be controlled through permissions configuration.

Using manual interventions, customer service representatives can create credit memos for refunds and debit memos for charges. Manual authorization and charge requests can also be made. Orders can be put in a payment held status and then released to continue processing. New payment methods can be added as needed. Existing payment methods can be suspended and re-activated. If the payment method is suspended, all the other open authorization and charge requests created for this payment method are closed, and no new authorization or charge requests are created for this payment method.

12.1.19 Multiple Control Levels

Payment processing can be enabled or disabled at the document type level. For example, if payment processing is disabled for the Planned Order document type, no payment related activities occur for any orders belonging to that document type. Furthermore, payment processing can also be controlled at the seller level. Payment processing can be disabled for specific sellers so that their orders do not carry out payment processing. The document type permission setting overrides the seller-level control setting.

12.1.20 Payment Rules

One or more payment rules can be configured for the Seller organization. One of these rules should be set as the default payment rule for the Seller organization. Either a payment rule applicable to the order can be specified when creating the order or the Seller's default payment rule is used for the order. The payment rule specifies whether payment processing is done through accounts receivable or Selling and Fulfillment Foundation, the type of payment processing to be done by the Selling and Fulfillment Foundation – Authorization or Settlement or both, and so on.

12.2 Payment Type Groups

Payment Type Groups refer to the method of tracking payment types that behave in the same manner. The available payment type groups are:

- Credit Card
- Customer Account
- Stored Value Card
- Other

12.3 Payment Types

Payment types refer to the method of tracking payment-related information used by the seller organization. The standard payment types provided are as follows:

- CREDIT_CARD
- CUSTOMER_ACCOUNT
- CHECK
- OTHER

A seller organization can define any number of custom payment types. All of the payment methods used in an order must be of payment types that have already been configured as valid payment types for that Seller.

For each payment type, the charging sequence and the refund sequence can be specified in the seller configuration. A charge sequence number (or refund sequence number) of 0 has the highest priority, followed by 1 and so on. Multiple payment types can have the same sequence number. If the sequence is not specified, all payment types are assumed to be of the same priority. The charge sequence and refund sequence are independent of each other.

12.3.1 Payment Methods

Payment Methods contain the payment information specific to a payment type and are sent down with an order. They may also contain payment details pertaining to pre-authorization or pre-charge information. Selling and Fulfillment Foundation supports any number of payment methods to be passed in with each order.

The following are the required fields for each payment type when it is used within a payment method in an order:

Table 12–3 Required Fields for Payment Type

Payment Type Group	Required Field
Credit Card	CreditCardNo
Customer Account	CustomerAccountNo
Stored Value Card	SVCNo
Other	PaymentReference1

One or more payment methods for an order can also be assigned charging sequence. If the charge sequence is not specified for the payment methods or at the seller configuration level, all payment methods in the order has the same priority with regard to charging. The charge sequence of the payment method is subordinate to the sequence defined at the seller configuration level.

A payment method can have unlimited charges assigned to it. If Unlimited Charges = 'N', a Maximum Charge Limit can be set on it. Authorizations and charges made on any payment method are subject to the limit set by the Maximum Charge Limit if Unlimited Charges = 'N' for that payment method. Any payment method having Unlimited Charges = 'Y' is assigned the remaining amount of authorization/ charge on the order and any payment method below it in terms of charge sequence is ignored.

A payment method in an order can be modified, suspended, or removed. New payment methods can also be added to an existing order. For detailed information, see the `changeOrder()` API in the *Selling and Fulfillment Foundation: Javadocs*.

12.3.2 Tokenization of Sensitive Information

By default, credit card numbers and stored value card numbers are tokenized instead of being encrypted. Tokenization is the process of storing sensitive information, such as a credit card number or stored value card number, in a vault system that associates a token to the securely-stored sensitive information.

The process of tokenization is enabled with the Sterling Sensitive Data Capture Server application. Sterling Sensitive Data Capture Server interprets, validates, and tokenizes credit card numbers and stored value card numbers, enabling the Sterling Selling and Fulfillment Suite to achieve Payment Application Data Security Standard (PA-DSS) compliance.

For additional information about the Sterling Sensitive Data Capture Server application and the tokenization of credit card numbers and stored value card numbers, refer to the *Sterling Sensitive Data Capture Server Configuration Guide*.

For information about how to configure the Sterling Sensitive Data Capture Server application for PCI-DSS compliance, refer to Sterling Commerce's *PA-DSS Implementation Guide*.

12.3.3 Charging Sequence

The charging sequence determines the sequence in which Selling and Fulfillment Foundation creates authorization or charge requests. The logic is as follows:

Each payment method on the order is assigned an Actual Charge Sequence based on which authorization and charge requests are created. The requests are created against the payment methods in the ascending order of the payment type's configured charge sequence as defined in the seller configuration and then the payment method's transactional charge sequence as defined in the specific order.

For example, in this scenario, the charge sequence for the seller is specified as follows:

- 0 for CUSTOMER_ACCOUNT
- 1 for CREDIT_CARD
- 2 for OTHER

The payment methods in the order are specified to use the following charge sequence:

- 1 for CreditCard1
- 2 for CustomerAccount1
- 3 for Other1

- 4 for CreditCard2

This results in these payment methods charged in the following sequence:

1. CustomerAccount1
2. CreditCard1
3. CreditCard2
4. Other1

Also note the following behavior regarding charging:

- The requests created do not exceed the Max Charge Limit specified on the Payment methods.
- If a particular payment method is not valid (information incomplete), the system skips that payment method and creates charges against lower priority payment methods in the order.
- Authorization and Charge requests are not created against suspended payment methods.
- If the payment methods on the order are not sufficient to cover the required funds, the order rolls back to the AWAIT_PAY_INFO status. At this point in the process, the PAYMENT_COLLECTION.INCOMPLETE_PAYMENT_INFORMATION event can be raised.

12.3.4 Refund Sequence

The refund sequence determines the sequence in which Selling and Fulfillment Foundation creates negative charge or refund requests.The logic it follows is described below.

1. The refund requests are created against the payment methods in the order of the 'Actual Refund Sequence'. The requests are created against the payment methods in the ascending order of the payment type's configured refund sequence as defined in the seller configuration and then the descending order of the payment method's transactional charge sequence as defined in the specific order.
2. The refund requests are created only against payment types that are marked as VALID_FOR_RETURN. If a payment method is suspended,

it is not considered for refund. The steps involved in determining the refund amount are:

a. In the first parse of the order's payment methods, the refund against a payment method does not exceed the total charge against the payment method.

b. After the first parse finishes, there could still be some refund that needs to be issued for which a valid payment method, that has the relevant funds charged against it, could not be found. For this extra refund amount, the following logic is applied:

- If the "PAYMENT->EXCEED_CHARGE_AMOUNT_FOR_REFUND" rule is set as Y (a document type or rule set level definition), the first priority (lowest Actual_Refund_Sequence) payment method on the order is chosen to issue all remaining credit.
- Even the above option might fail if there are NO payment methods on the order that are valid for return. In this case, the payment Type, marked as Default For Return, is created for the order and the refund is issued against this payment type. If this payment type requires more information to make it complete, an INCOMPLETE_PAYMENT_INFORMATION event is raised.

If both the above fail (there is no Default_For_Return payment type), then an INCOMPLETE_PAYMENT_INFORMATION event is raised. For more information about refund fulfillment, see Section 12.1.16, "Returns Payment Processing and Refund Fulfillment".

12.4 Payment Status

Each order has a payment status, such as AUTHORIZED, INVOICED, or PAID. This status is associated with the order. Payment status can have one of the following values:

- AWAIT_PAY_INFO - The payment information was not available at order creation or during authorization or charging. An order can also go to this state if the provided payment methods are insufficient to cover the order amount or the amount to be refunded.
- AWAIT_AUTH - Part of the order amount is pending authorization.
- REQUESTED_AUTH - The authorization request has been sent, but a reply has not been received from the payment system. This status only occurs in asynchronous environments.

- REQUESTED_CHARGE - A charge request has been sent, but a reply has not been received from the payment system. This status only occurs in asynchronous environments.
- AUTHORIZED - The order amount is less than or equal to the authorized amount. When delayed reauthorization is configured, a status of AUTHORIZED indicates that the order can move through the pipeline, but may not indicate full authorization. See Section 12.5.1.4, "Delayed Reauthorization" for more information.
- INVOICED - The order is invoiced completely and there is no open order amount.
- PAID - The payment was collected for the order.
- FAILED_AUTH - An authorization request failed.
- FAILED_CHARGE - A charge request failed.
- NOT_APPLICABLE -Either the document type of the order does not require payment processing, or the seller organization does not require payment processing.

12.5 General Payment Process

Payment processing in Selling and Fulfillment Foundation consists of two sub-processes - Authorization and Settlement.

12.5.1 Authorization Process

Payment Authorization is a process through which the amount to be paid on a payment method is verified. In case of credit cards, authorization specifically involves contacting the payment system and blocking the required amount of funds against the credit card. Payment types may or may not require this authorization step. This is configurable in Selling and Fulfillment Foundation in the sellers payment rule. Note that if an order requires payment processing, Selling and Fulfillment Foundation does not pick up the order for scheduling or other processing until it is authorized.

The Payment Collection time-triggered transaction analyzes an order to create authorization requests. The Payment Execution time-triggered transaction monitors requests created for authorization and provides user

exits to carry out the authorization. The user exit can process the authorization request in any one of the following ways:

- Perform synchronous processing to carry out the authorization immediately by interfacing to an accounts receivable database, and pass back the authorized amount.
- Place a request to try again later if the interface to the payment system is inoperable.
- Request asynchronous processing, which means that Console never contacts the payment system for this order.

Depending on the response from the external payment system, different events can be raised to handle the response appropriately within Selling and Fulfillment Foundation. For more details on the available events, see the payment processing APIs in the *Selling and Fulfillment Foundation: Javadocs*.

Once authorization is received, or the order is pre authorized on the front end for the complete order amount, the Payment Collection transaction changes the payment status to AUTHORIZED.

12.5.1.1 Expiration Date

Each authorization has an attached expiration date. At a fixed number of days (=Expiration for Authorization, which is configurable) before the authorization expires, a reauthorization request is automatically created by the Payment Collection time-triggered transaction. For example, if an order expires on 4/15, and the fixed number of days is 4, then the reauthorization request is created on 4/11.

12.5.1.2 Charge Transaction Request Identifiers

The charge transaction request identifiers configuration option, which is accessible from the Payment Rule screen, provides customers with a finer granularity of authorization control on an order. Instead of payment authorizations being processed at only the order level, this configuration option enables payment authorizations to be based on predefined order groupings. Each grouping in the order has a charge transaction request identifier associated to the grouping.

Note: The charge transaction request identifiers feature does not support the Settlement Required payment option.

Two hang-off tables, YFS_CHARGE_TRAN_REQUEST and YFS_CHARGE_TRAN_RQ_MAP, manage the grouping requests of authorization pertaining to an order. YFS_CHARGE_TRAN_REQUEST contains the requests for authorizations and identifiers, and YFS_CHARGE_TRAN_RQ_MAP maps the requests to charge transactions.

12.5.1.2.1 Sequence of Authorization Requests

Each authorization request contains a MaxRequestAmount attribute, which defines the total authorization required for that request. When a request is fully authorized, the authorization for the next request begins. A sequence may be specified for the requests; otherwise, the requests are executed in the order of their request identifiers.

12.5.1.2.2 Payment Status

Each authorization request has a payment status associated with the request. The payment status can have a value of AWAIT_AUTH, REQUEST_AUTH, FAILED_AUTH, or AUTHORIZED. The default payment status is AWAIT_AUTH. When the payment status of a request identifier changes, the PAYMENT_COLLECTION.REQUEST_PAYMENT_STATUS event is fired.

The payment status on a request is computed in the same way as the payment status on an order.

12.5.1.3 Reversal of Authorization

In addition to normal transaction fees that credit card companies charge, they may also charge an extra fee to merchants who do not explicitly reverse an unused credit card authorization. The Reverse Authorization feature enables merchants to implement an authorization strategy that generates a reversal request before the unused authorization expires.

Credit card companies may also levy a fee if the authorization amount and the settlement amount do not match. Selling and Fulfillment Foundation provides a means of reversing expiring authorizations and handling different authorization and settlement amounts through the YFSCollectionCreditCardUE user exit implementation. Reverse authorization requests are processed by the PAYMENT_EXECUTION agent.

With the exception of customer accounts, the reverse authorization feature supports any payment method that requires authorizations.

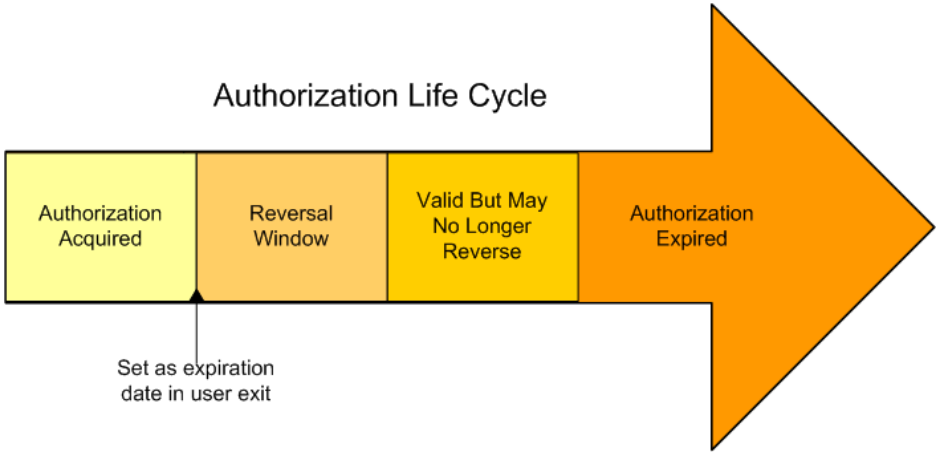
12.5.1.3.1 Authorization Reversal Window

During the credit card authorization process, a merchant seeks authorization for an order by contacting a payment system and then blocking the credit card for the settlement amount. Typically, as this authorization expires, it automatically seeks reauthorization. However, if reverse authorization is enabled, in addition to seeking reauthorization, an authorization request is created for the same authorization ID, only in a negative amount. For example, an expiring authorization for $100 would initiate another authorization for $-100.

While the credit card company and the bank influence when the expiration occurs and when the reversal period ends, you can define an authorization reversal window based on your business metrics. These metrics might include the volume of throughput you have to the payment system, how often order modification is permitted, the percentage of orders against future inventory, and how often you run payment collection agents.

Figure 12–1 illustrates a typical authorization life cycle and where the reversal window might be in that life cycle.

Figure 12–1 Authorization Window of Reversal



In Figure 12–1, a reverse authorization can occur any time from the start of Authorization Acquired to the end of the Reversal Window. The ideal point of reversal for an authorization is as close to the end of this

reversal period as possible, so that you can head off extra fees, but not too early in this reversal period, as this will incur additional costs.

If you are reversing authorizations, set the expiration date in the YFSCollectionCreditCardUE to the point shown in Figure 12–1, not the actual authorization expiration date. As a result, the system will start trying to reverse the authorization at this point, when there is still time to reasonably process a reverse authorization and to retry if the first attempt fails.

If you are not reversing authorizations, set the expiration date to the actual expiration date.

In some implementations and for some credit cards, the reversal window may differ from the one depicted in Figure 12–1. The period "Valid But May No Longer Reverse" may not exist because it might be possible to reverse an authorization right up to the actual expiration date without incurring extra fees.

12.5.1.3.2 Configuration Options

Two reverse authorization configuration options are described in this section. These are available in the Payment Type Details window of the Distributed Order Management Applications Manager, as described in the *Sterling Distributed Order Management: Configuration Guide*:

- Do Not Reverse
- Reverse When Expired

Do Not Reverse

You can select this option if you do not want to implement an authorization reversal strategy. This means that the normal authorization life cycle occurs; authorizations are acquired, then after a period of time, they expire. At expiration, new authorizations are acquired. No reversals are involved.

Reverse When Expired

If this option is enabled, reverse authorization occurs automatically in the form of an authorization request for the same authorization ID, only in a negative amount. For example, if the original authorization was for $60, the reverse authorization request is for -$60.

If you have enabled automatic reversal of authorization and the specific payment method does not support reverse authorization, Selling and Fulfillment Foundation marks the authorization for reversal but does not contact the payment system and incur any cost.

12.5.1.3.3 Reverse Authorization Settlement Scenarios

This section describes three settlement scenarios for reversal of authorization.

- Authorization Amount and Settlement Charge Match
- Expired Authorization
- Authorization Amount and Settlement Charge Differ

Authorization Amount and Settlement Charge Match

In most scenarios, a settlement request will be created for an authorization that has not yet expired and for the exact amount. In this situation, there is no need for reversal or reauthorization, and the settlement can occur normally against the intended authorization.

Expired Authorization

It is possible for a charge request to be created while an authorization request has expired, but before it can be reauthorized. In this instance, there is no need to reverse the authorization. If the authorization expiration date is set to the last date when reversal is allowed, then the authorization is likely still valid for settlement.

You can use executeCollection's user exits such as YFSCollectionCreditCardUE to determine how the charge should be processed. You can determine the actual authorization date based on the payment type information indicating how far in advance a payment type must be reversed.

If the authorization is actually expired, then the YFSCollectionCreditCardUE can return that the charge has failed and requestCollection would create a new charge request, returning the results in executeCollection. Inside the output of the user exit is a field in recordAdditionalTransactions that you can use to record additional payment system communication that occurs inside the YFSCollectionCreditCardUE user exit, which typically updates only a single record. The executeCollection API will use this document to call the

recordExternalCharges API internally. If the reauthorization or settlement fails, then YFSCollectionCreditCardUE returns that the charge has failed and requestCollection will create a new charge request.

Authorization Amount and Settlement Charge Differ

When settling an invoice, an authorization request may not match the actual charge request. This mismatch can occur if there is a partial shipment, a price change, or other adjustment scenarios. For example, a charge for $60 might take place on an authorization for $100. If this happens, use the following strategy:

1. Reverse the original authorization of $100.
2. Acquire a new authorization for $60.
3. Settle against the new authorization for $60.

You can implement this strategy inside the YFSCollectionCreditCardUE. and report any necessary information via the recordAdditionalTransactions output field. You may also update the authorization ID of the charge record, but you must provide reverse and reauthorization information using recordAdditionalTransactions.

12.5.1.3.4 Manual Adjustment of Authorizations

You may choose to minimize mismatched settlements by reversing authorizations for order modifications and invoices. Depending on the frequency of total order modifications, this may save transaction fees. Tax and charge adjustment during invoicing can also be a candidate for reversal and reauthorization, as these occur right before settlement.

To accomplish manual adjustment, an open authorization reversal request can be created using the recordExternalCharges API; a reauthorization should not be created at this time. With the reversal created, the payment agents will create the reauthorization automatically for the correct amount.

12.5.1.4 Delayed Reauthorization

When an order is awaiting inventory, authorizations can expire and trigger multiple reauthorizations that lock up the customer's credit line and cause expense to the seller. To reduce the cost of these transactions, configurable payment rules offer the following delayed reauthorization choices:

- **Delay of Reauthorization** - Delay reauthorization after the initial authorization has been created. When the initial authorization expires, no reauthorizations occur until a configurable number of hours before the release date.
- **Schedule without Authorization** - Forego the initial authorization and wait until a configurable number of hours before the release date to perform an authorization.

Both configuration options are accessible on the Payment Rule screen.

12.5.1.4.1 Release Date

The release date, which is when notice is sent to the warehouse to release the order, drives the timing of the final authorization. This is not the same as the ship date, when an order ships. Scheduling a final authorization hours before the release date guarantees that lines will not be released or shipped until authorization takes place.

12.5.1.4.2 Bundles

Bundle parents will be priced for authorization similar to Invoicing when the "Invoice on first bundle component" rule is selected. For more information on Invoicing, see the *Sterling Distributed Order Management: Configuration Guide*.

12.5.1.4.3 PS and DS Work Orders

Service lines use the Promised Appointment Start Date, rather than the release date, for determining the authorization date. Product items on a work order still use the release date.

12.5.1.4.4 Procurement, Transfer, and Chained Orders

Procurement, Transfer, and Chained Orders can begin without authorization.

12.5.1.4.5 Order Modifications

Any modifications that affect the payment related entities on an order cause the order payment status to change to AWAIT_AUTH. Examples of these changes include changing a payment method, increasing line quantity because of adding a new line or increasing the quantity on a line, and changing a price, charges, and taxes. After these modifications,

the Payment Collection transaction evaluates the order again and determines the appropriate course of action.

When delayed reauthorization is configured, order modifications include any transaction or API that:

- Changes the price of an order (except shipment adjustments and final invoices on a shipment)
- Changes the expected ship date
- Changes the expected appointment date of PS/DS lines

12.5.1.4.6 Payment Status

If authorization is required before scheduling, the entire order that is not covered by an active authorization will be reauthorized. If the payment rule requires authorization before scheduling and the price change is positive (increases), the payment status becomes AWAIT_AUTH.

If authorization is not required before scheduling, the payment status will remain AUTHORIZED until the order is ready for release.

Note: When delayed reauthorization is configured, a payment status of AUTHORIZED on an order awaiting inventory may not indicate full authorization. For example, if a $100 order for Item 1 and Item 2 is initially authorized, but Item 1 ships immediately for $60 and Item 2 is delayed by three weeks, the initial AUTHORIZED status for $100 does not necessarily constitute authorization for the remaining $40 after the authorization expires. This AUTHORIZED status does, however, keep the item or order moving through the order process until release, when final authorization occurs.

12.5.2 Settlement Process

Payment settlement involves collecting the funds for the amount recorded for an order. For example, when using credit cards, the settlement process specifically involves contacting the payment system and collecting the required amount of funds against the credit card. Payment types may or may not require this settlement step. This is

configurable in Selling and Fulfillment Foundation in the seller's payment rule. Note that if an order requires payment processing, it is not purged unless settlement is completed and order is in PAID status. For more details on the available events, see the payment processing APIs in the *Selling and Fulfillment Foundation: Javadocs*.

Payment Collection analyzes orders to create settlement requests. A settlement request is a request for Selling and Fulfillment Foundation to complete a settlement. If the payment rule applicable for the order requires both authorization and settlement, the settlement requests are created only against existing, non-expired authorizations.

If settlement is handled through an external collections system, you may choose not to record payment collections through accounts receivable. If settlement is not recorded in the Selling and Fulfillment Foundation database, a settlement request is not created for the order. It is assumed that settlement is performed through an external system using the information published with the invoice.

Payment Execution monitors settlement requests. This time-triggered transaction provides user exits to carry out the settlement. The user exits can process the settlement request in one of the following ways:

- Carry out the settlement immediately (interfacing to an accounts receivable database) and pass back the settlement amount.
- Place a request to try again later if the interface to the payment system is inoperable.
- Request asynchronous processing, which means that Console never contacts the payment system for this order. The payment interface (custom extension) must update details corresponding to payment through the Selling and Fulfillment Foundation APIs. This is typically the case when interfaces to the payment system are run in batch mode.

The Selling and Fulfillment Foundation time-triggered transactions change the payment status of the order to PAID once payment is received for the complete order amount.

12.6 Customer Account Payment Process

When customer accounts are set up within the system, payment processing for orders belonging to these customer accounts differs from that for the General Payment Process.

A customer account can be set up in the system by setting the Customer Account Maintained Internally flag for the payment rule, and setting the account limit for that customer account. For this functionality to work effectively, the Hold To Be Applied Due To Insufficient Funds In Customer Account rule should be configured. Besides this, the deprecated functionality of hold types should not be on. For more information about hold type, see Section 7.8, "Order Hold Processing".

12.6.1 Authorization Process

When a customer account is maintained, neither is the external system contacted for authorization nor is the YFSCollectionCustomerAccountUE user exit called for AUTHORIZATION requests. The system checks how much funds are available in the customer account. If the requested amount is less than or equal to the available amount in the customer account, the AUTHORIZATION request is closed and the Open Authorization amount against that customer account is increased. If the requested amount is unavailable, the hold configured in the Applications Manager, Hold To Be Applied Due To Insufficient Funds In Customer Account is applied to the order. The requested amount is added back to the customer account for negative authorization requests.

12.6.1.1 Payment Processing In Case of Insufficient Funds

Orders are on hold when there are insufficient funds in the account. The hold is determined by the rules configured in the system. When the orders that are on hold gets authorized, the hold is removed.

To make the orders that are on hold eligible for reauthorization:

- Manually Override Authorization

  In exceptional scenarios, the order is processed even if sufficient funds are not available. The book amount for the customer account is increased and the total amount of the order is considered as processed amount.

- Change Customer Account Limit

A customer may request the enterprise to increase the account limit. When the customer account limit is increased, the orders that could not be processed earlier should be revisited.

The triggerRequestCollection API is provided to trigger payment processing of such orders that are on hold. This API should be called on change of customer account limit.

- Reduce Open Order Amount

  The Open Order Amount for a customer account may get reduced when an authorized order for that customer account is cancelled, or when an order is settled either internally or externally.

  The triggerRequestCollection API is provided to trigger payment processing of such orders that are on hold. This API should be called on the success of the change order.

- Settlement of Customer Account Orders

  When customer account orders are settled, the execution of open orders occurs.

  The triggerRequestCollection API is provided to trigger payment processing of such orders that are on hold. This API should be called on change of customer account balance.

12.6.1.2 Expiration Date

The authorization acquired for customer accounts do not have an expiration date.

12.6.2 Settlement Process

The payment settlement process involves collecting funds for the amount recorded for an order. The settlement process for orders pertaining to the customer accounts maintained internally is almost the same as that for the general payment processing settlement process. For more information about General Payment Process, see Section 12.5.2, "Settlement Process". The only difference in the settlement process for internally maintained customer accounts is that the open order amount on those accounts gets reduced by the settled amount.

12.7 Payment Processing Transactions

The following standard time-triggered transactions enable Selling and Fulfillment Foundation to exchange financial information with external systems:

- Create Order Invoice - Creates one or more invoices for an order. It is a derived transaction and is recommended to be used in scenarios where invoices are not linked to shipments (such as Return).
- Create Shipment Invoice - Creates one or more invoices for the shipment. It is a derived transaction and is recommended to be used to create invoices against shipments.
- Payment Execution - Processes all requests that are pending authorization and charging. It reads all open requests for authorization and charges and invokes the appropriate user exits for executing the request.
- Payment Collection - Analyzes the orders and determines the amount for which an authorization or charge request(s) should be created.
- Send Invoice - Publishes invoice data that can be directed to external accounts receivable systems. The invoice can be published when it is created or after the payment has been collected.

For more information about these time-triggered transactions, see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide*.

12.8 Using Custom Transactions

If your business needs require additional financial-related integration with external systems, you can write custom time-triggered transactions. When writing custom time-triggered transactions, invoke any of the Selling and Fulfillment Foundation payment-related user exits and APIs.

12.8.1 Payment-Related APIs

The following APIs enable integration with external financial systems:

- executeCollection() - Processes individual requests for authorization and charging in the order the requests were created. If you need to process requests individually, you can call this API to carry out

individual requests as required. This API calls the appropriate user exit to carry out payment processing.

- recordCollection() - Records the authorization and charging amounts processed for individual requests created by Selling and Fulfillment Foundation. This is useful when you want to interface with external payment processing systems in a batch mode to process a group of requests and then update Selling and Fulfillment Foundation with the results.
- processOrderPayments() - Invokes the requestCollection() API and executeCollection() API in a single call, which allows for online authorization of an entire order amount in order to avoid unnecessary expenses. The processOrderPayments() API takes a list of payment methods on the order.
- executePaymentTransactions() - Processes requests for authorization and charging before creating or confirming an order. Multiple payment requests are authorized or charged in the order the requests are made. This API verifies payment conditions before calling the appropriate user exit to carry out payment processing. If the API encounters an error during verification, it terminates the process without calling the user exit. If an error occurs while the user exit is processing, the API terminates the user exit and returns all successful payment transactions up to and including the first failed transaction.
- scheduleOrder() - Calculates when orders should be reauthorized.
- getOrderDetails() - Returns detailed information about a specific order number from the transaction or history tables. Information includes `OrderLines`, `PriceInfo`, `OverallTotals`, `ChargeTransactionDetails`, `OrderStatus`, and other order-related information.
- recordExternalCharges() - Records any authorizations and charging carried out external to Selling and Fulfillment Foundation. This API is also used to record a payment received by an external system. An example is a buyer account that receives payment after a considerable period of time. The executeCollection API would have previously notified Selling and Fulfillment Foundation that this charge is being processed asynchronously, then the recordExternalCharges API records the actual charge.

- requestCollection() API - Analyzes the order and determines the amount for which an authorization or charge request should be created.
- recordInvoiceCreation() - Records credit memos, debit memos or informational invoices against an Order. When a credit memo or debit memo is recorded using this API, it acts as a trigger for Selling and Fulfillment Foundation to invoke settlement processes for the amount being invoiced. Selling and Fulfillment Foundation does not do any payment processing for informational invoices created through this API. These invoices can be used to capture carrier claims and so forth which are actually processed through an external system but are recorded against the order for visibility purposes.
- createAccessToken - Generates a random and unique access token. This is used to authenticate calls to the Sterling Sensitive Data Capture Server application.
- manageChargeTransactionRequest and getChargeTransactionRequestList - These APIs manage the authorization process for the charge transaction request identifiers associated with an order. The getChargeTransactionRequestList API retrieves the charge transaction requests for authorization. The manageChargeTransactionRequest API also resets the authorization expiration date on the order to sysdate, which, in turn, enables the requestCollection API to determine the amount for which the authorization or charge request should be created. Each API call can simultaneously process multiple charge transaction requests for an order.

For more information about these APIs and the events raised by them, see the *Selling and Fulfillment Foundation: Javadocs*.

12.8.2 Payment-Related User Exits

The following user exits are called by the Payment Execution time-triggered transaction and the executeCollection() API:

- YFSCollectionCreditCardUE - Processes authorization and charging against a credit card.
- YFSCollectionCustomerAccountUE - Processes authorization and charging against a buyer account.

- YFSCollectionOthersUE - Processes authorization and charging against any other method of payment such as gift checks or discount coupons.
- YFSgetFundsAvailableUE - Enables a user to plug in external logic to determine the funds available on a payment method for an order.
- YFSCollectionStoredValueCardUE - Provides the option of plugging in custom logic to authorize or charge a stored value card.

12.9 Database Details

The Selling and Fulfillment Foundation database holds payment data mainly in six tables, YFS_CHARGE_TRANSACTION, YFS_CHARGE_TRAN_DIST, YFS_ORDER_HEADER, YFS_CHARGE_TRAN_REQUEST, YFS_CHARGE_TRAN_RQ_MAP, and YFS_PMNT_TRANS_ERROR.

The system uses the YFS_CHARGE_TRANSACTION table as the driver table to interface with the time-triggered transactions for processing authorizations and charges. The driver table also serves as the journal for all credits and debits against the order at any time.

YFS_CHARGE_TRANSACTION Table

The key fields and their values in the YFS_CHARGE_TRANSACTION table are detailed below.

Table 12–4 YFS_CHARGE_TRANSACTION Table Fields

Field	Description
STATUS	This field has the following valid values: • OPEN - A new request is created in the YFS_CHARGE_TRANSACTION table. These records are picked up by the Payment transactions. • CLOSED - Payment collection was performed external to Selling and Fulfillment Foundation and reported collection details through the recordCollection() API or payments were authorized or collected using the Payment Execution Time-triggered Transaction. • CHECKED - The record in YFS_CHARGE_TRANSACTION table has been validated. No further processing occurs on these records. • ERROR - Authorization or charging of the associated payment method failed.

Table 12–4 YFS_CHARGE_TRANSACTION Table Fields

Field	Description
CHARGE_TYPE	This field has the following valid values: • ADDITION - Ordered Quantity of an order line has increased. • ADJUSTMENTS - There is mismatch between user-specified TotalAmount of an order and the calculated TotalAmount. • AUTHORIZATION - An authorization is requested. The request can be created by calling the requestCollection API or it can be created through the Payment Collection time-triggered transaction. • CANCEL - An order or part of an order was cancelled. • CHANGE_PRICE - An order has been repriced. An order can be repriced by changing unit price, charges, and taxes. • CHARGE - A charge is requested. The request can be created by calling the requestCollection API or it can be created through the Payment Collection time-triggered transaction. They place such requests for open orders or shipments. • CREATE_ORDER - An order was created in the system. • ORDER_INVOICE - An order has been partially or completely invoiced, using CREATE_ORDER_ INVOICE transaction. • REQUEST_SETTLEMENT - Ordered Quantity (partially or completely) of an order line has been settled before invoicing. This occurs during pre-settlement requests. • RETURN - Return order is partially or completely invoiced.

Table 12–4 YFS_CHARGE_TRANSACTION Table Fields

Field	Description
	• SHIPMENT - An order or part of an order has shipped and then invoiced using the Create Shipment Invoice time-triggered transaction. • SHIPMENT_ADJUSTMENT - This charge request is created if any adjustment is needed after an order is completely shipped. • SPLIT_LINE - Order line is split and the new line requires payment processing. • TRANSFER_IN - This record is displayed when amounts have been transferred in from another order. Clicking on this link takes the user to the order the amounts have been transferred from. • TRANSFER_OUT - This record is displayed when amounts have been transferred to another order. Clicking on this link takes the user to the order the amounts have been transferred to.
CREDIT_AMOUNT	The amount credited to the customer against the associated payment method for the order. The amount reflects funds collected from the customer. It increases only after actual fund collections have taken place.
DEBIT_AMOUNT	The amount debited to the customer against this order.
BOOK_AMOUNT	The open order amount. At any point, it reflects the total order amount that has not yet shipped. The amount is increased by order creation, addition of Ordered Quantity, addition of lines, increase in price, charges and taxes and reduced by cancellations, reduction in prices, charges and taxes, and invoicing.
OPEN_AUTHORIZED_AMOUNT	The amount authorized against the customer's payment method. The authorizations are used for actual fund collection.
REQUEST_AMOUNT	The amount for which an authorization or charge request has been made.

Table 12–4 YFS_CHARGE_TRANSACTION Table Fields

Field	Description
SETTLED_AMOUNT	The amount for which a pre-settlement request has been made.
USER_EXIT_STATUS	This field gets updated to 'INVOKED' when Payment Execution transaction picks up records for processing. While waiting on processing from an external payment system, this field is updated to 'ONLINE', which prevents agents from picking up the record. It is set back to <blank> when the transaction returns successfully. If this field remains populated, it indicates that a problem occurred with the external system call and requires manual intervention.

The following table describes the various amounts logged in the YFS_CHARGE_TRANSACTION table at different points in the order life cycle.

Table 12–5 Payment Charge Amounts within the Order Life Cycle

When	Charge Type	Credit Amount	Debit Amount	Book Amount	Authorized Amount	Requested Amount
Creation of Order	ORDER_CREATE			+ TOTAL AMOUNT on order		
Authorizations performed on front-end and passed with CreateOrder XML	AUTHORIZATION				+ Authorized Amount as passed	
Funds collected on front-end and passed with CreateOrder XML	CHARGE	+ Charge Amount as passed				
Cancellations	CANCEL			- Cancel Amount as		

Table 12–5 Payment Charge Amounts within the Order Life Cycle

When	Charge Type	Credit Amount	Debit Amount	Book Amount	Authorized Amount	Requested Amount
Shipments when payment collection is recorded in the Console database	SHIPMENT		+ Shipment Amount	- Shipment Amount		
Shipments when payment collection is not recorded in the Console database	SHIPMENT	+ Shipment Amount	+ Shipment Amount	- Shipment Amount		
Creation of authorization requests	AUTHORIZATION					Authorization amount
Creation of charge requests	CHARGE					Charge Amount
Actual Authorization					Amount Authorized	
Actual Fund Collection	CHARGE	Amount Collected				
Reversal of Authorization	AUTHORIZATION			- Amount Authorized		
Price Change through API/ Console	CHANGE_PRICE			Price Change Amount		
Debit Memo	ADJUSTMENT		Amount on Memo			
Credit Memo	ADJUSTMENT		Negative amount on the memo			

Table 12–5 Payment Charge Amounts within the Order Life Cycle

When	Charge Type	Credit Amount	Debit Amount	Book Amount	Authorized Amount	Requested Amount
Return	RETURN		Negative of the total credit given to customer			
Ordered quantity of line(s) increases	ADDITION			Additional amount		
Order is invoiced	ORDER_INVOICE		Invoiced Amount	Negative of Invoiced amount		
Order line is split	SPLIT_LINE			Change in Total Amount		
Creation of Last Shipment	SHIPMENT_ ADJUSTMENT			Difference between order book amount and total invoiced amount		
Creation of Return Order	TRANSFER_IN	Amount transferred to order				
Creation of Return Order	TRANSFER_OUT	Negative of amount transferred to order				
Deferred Credit	DEFERRED_ CREDIT	Amount Deferred				

YFS_CHARGE_TRAN_DIST Table

This table contains records for pre-collected funds that are being refunded. It also contains the keys of the pre-collected charge

transactions (transfer in, charge), the charge transactions to which it was refunded (transfer out, negative charge), and the amount refunded.

Table 12–6 YFS_CHARGE_TRAN_DIST Table Fields

Field	Description
DISTRIBUTED_FROM_CT_KEY	The charge transaction that is associated with this charge transaction distribution. This is the charge transaction that distributed the funds.
DISTRIBUTED_TO_CT_KEY	The charge transaction that is associated with this charge transaction distribution. This is the charge transaction that funds were distributed to.
CHARGE_AMOUNT	The amount that was distributed to the charge transaction record, DISTRIBUTED_TO_CT_KEY, from the charge transaction record, DISTRIBUTED_TO_CT_KEY.

YFS_ORDER_HEADER Table

The key field related to payment processing in the YFS_ORDER_HEADER table is PAYMENT_STATUS. See Section 12.4, "Payment Status".

YFS_CHARGE_TRAN_REQUEST Table

This table contains the requests for payment authorization and identifiers. The combination of the ORDER_HEADER_KEY and the identifier creates a set of unique groups for the order. Authorizations will be split such that each authorization is for only one identifier.

Table 12–7 YFS_CHARGE_TRAN_REQUEST Table Fields

Field	Description
ORDER_HEADER_KEY	The order the request is for.
CHARGE_TRAN_REQUEST_ID	The unique identifier for a request of the amount on an order.
PAYMENT_STATUS	Similar to the payment status in the YFS_ORDER_HEADER table.

Table 12–7 YFS_CHARGE_TRAN_REQUEST Table Fields

Field	Description
REQUEST_SEQUENCE	The order in which the groups are authorized. Records with REQUEST_SEQUENCE not `null` are processed first; `null` sequences are ordered by CHARGE_TRAN_REQUEST_ID, in ascending order.
MAX_REQUEST_AMOUNT	The maximum amount this unique identifier represents before the distribution to another group begins to become Authorized.

YFS_CHARGE_TRAN_RQ_MAP Table

This table maps the charge transaction requests to the charge transactions filling them.

Table 12–8 YFS_CHARGE_TRAN_RQ_MAP Table Fields

Field	Description
REQUEST_AMOUNT	The amount requested on the mapped charge transaction.
PROCESSED_AMOUNT	The amount processed on the mapped charge transaction.

YFS_PMNT_TRANS_ERROR Table

This table contains a set of error messages obtained from the user exit output during payment processing. When records for a charge transaction are removed, any transaction error associated to the charge transaction must also be removed. This applies to the changeOrder() API, deleteOrder() API, and the PurgeOrderAgent.

Table 12–9 YFS_PMNT_TRANS_ERROR Table Fields

Field	Description
PMNT_TRANS_ERROR_KEY	The unique identifier that is associated with this payment transaction error record.
CHARGE_TRANSACTION_KEY	The charge transaction that is associated with this payment transaction error message. Each charge transaction may have multiple messages.

Table 12–9 YFS_PMNT_TRANS_ERROR Table Fields

Field	Description
MESSAGE_TYPE	The type of error message for this payment transaction error record.
MESSAGE	The error messages for this payment transaction error record.

13

Logistics Management

Logistics Management involves planning and monitoring order line level shipping processes so that an order is shipped when and how a customer wants it to be shipped. Selling and Fulfillment Foundation provides a number of features that support inbound compliance with buyer's shipping requirements, consideration of the preferences of an Enterprise, and the ability to customize the shipping processes.

13.1 Planning Considerations for Shipping

When planning to ship one or more order lines, the goals include:

- Delivering the items in a cost-effective way
- Meeting any requirements of the customer for shipping. For example, ensure that orders from different departments within the same organization are shipped individually to each department
- Deliver the items in a timely fashion

In practice, creating an effective plan for shipping depends on a large number of practical considerations. Often, there is a trade-off between the speed of delivery and the cost of delivery. The enterprise which is shipping items often attempts to minimize the number of shipments and loads that are sent, while the buyer is concerned with receiving shipments in a form that is convenient for the buyer's internal processes.

With Selling and Fulfillment Foundation, you can establish how these shipping planning considerations should be assessed. Using Selling and Fulfillment Foundation, you can increase the efficiency of your delivery planning by establishing conditions and processes for interacting with your buyers.

13.2 Terms

The following are some of the key terms used in shipping planning.

Carrier

The *carrier* is provider of the delivery service. Multiple carriers can be involved in delivering items that are part of an order.

Delivery plan

A *delivery plan* is a complete sequence of movements needed to deliver one or more orders from one or multiple origins to one or multiple destinations. A delivery plan is comprised of shipments, loads, origins, stops, and destinations.

Destination

The *destination* is the last node or address in the travel route of a load where all remaining shipments in the load are dropped off.

Load

A *load* carries one or more complete shipments (never a partial shipment) between two points. A load has one origin and one destination, but it can have multiple intermediate stops. Shipments can be added to a load at its origin or any intermediate stop and can be dropped off at the load destination or any intermediate stop.

Origin

The *origin* is the node the load originally ships from.

Shipment

A *shipment* is a delivery of one or more orders and order lines from a single shipper to a single consignee. A shipment can be carried through multiple loads and by multiple carriers.

Stop

A *stop* is any location where a shipment is picked up or dropped off. A load has a stop sequence that determines its travel route.

13.2.1 Shipment Planning Strategies

When planning, you want to deliver items cheaply, quickly, and honor the requirements of your customer. Some of the strategies used to accomplish these delivery planning goals are:

13.2.1.1 Consolidation of Items into a Shipment

It is often more efficient to ship several items from an order in one shipment. If all of the items are being shipped to the same location, it often cheaper to ship them together in a single shipment.

13.2.1.2 Consolidation of Shipments into a Load

Shipments can be batched together in a single load that is transported by the carrier. When consolidating shipments into a load, many factors should be considered:

- possible routes
- available carriers
- freight capacity of the carriers
- special requirements of either intermediate nodes or the customer

For example, there could be a low cost bulk carrier available, but the containers that are used by that carrier are too large for the receiving dock of the customer.

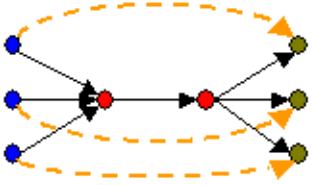
13.2.1.3 Optimize routing

Depending on the circumstances, different routing strategies can be used to optimize delivery time and delivery costs. In some situations, a direct route from the sender is best. In other cases, picking up multiple shipments, and delivering them to single destination makes the most sense.

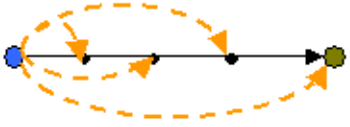
The following figure illustrates some of the most common delivery route scenarios, to give you a sense of the wide range of routing strategies that can be used.

Figure 13–1 Delivery Route Scenarios

Model 1 – Single Shipment, Single Load.
For simple shipments.

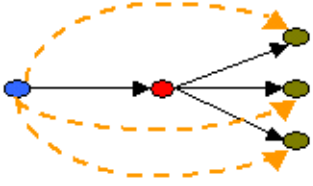


Model 2 – Many Shipments, Consolidate & Break Into loads
Allows multiple origins to consolidate (for example, at port) and be moved by second transport as a single load. Then broken back into initial structures to complete separate voyages.

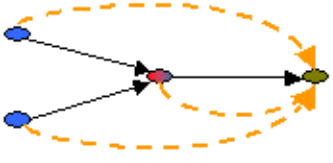


Model 3 – Many Shipments, Single load
Records a single load that makes stops to drop-off x amount of shipments.

Model 4 – Many Shipments, Intermediate Drop-Off & Pick-Up
Allows efficient use of available transport resources to create loads in transit.



Model 5 – Single Origin, Multiple Shipments, Break
Shipments are shipped from an origin as a single load and subsequently broken at 'nearest' point to different destinations. Separate loads make final delivery.



Model 6 –Single Shipment, Merge at one Origin
Example of a merge in transit where components are sourced from disparate sites and merged at the source of the high value component.

Model 7 – Single Shipment, Interline loads
Although a single origin/destination pair, the load, for cost/optimum performance reasons, moves with more than one carrier

13.3 Shipment Planning Features

Selling and Fulfillment Foundation provides many features to provide flexible shipment planning. You can establish conditions that control how consolidation of shipments, creation of loads and routing are performed. These conditions are established for the Enterprise, and can also be established for each Buyer organization. You can then establish the conditions of the organization that should take precedence if there are any conflicting conditions.

This section introduces the key features used in shipping planning. Key delivery planning features include:

- Using Economic Shipping Parameters
- Determining Routing

In addition, you can explicitly establish how your shipments are routed, using manual routing. See Section 13.3.2.3, "Delivery Plans".

13.3.1 Using Economic Shipping Parameters

Consolidating two or more shipments and shipping them together strengthens the potential to bring about significant savings in transportation cost. This also brings about a trade-off between the cost savings made through consolidated orders and the ability to stick to the delivery dates.

Economic Shipping Parameters (ESP) settings are used to assess the said trade-off between:

- Holding a shipment until many items are available to ship together, and
- Shipping in a timely fashion

Shipments can be held in two ways:

1. Multiple orders are available at a given time, and the orders are consolidated into a single shipment to reduce costs
    a. Multiple orders with the same ship date
    b. Multiple orders with different ship dates
2. An existing order is held in anticipation of another order to enable consolidation

Scenario 1-A: Multiple orders available at a given time with the same ship date

This is an ideal scenario where all the available orders can be consolidated into one shipment, without compromising the delivery dates.

This is possible as all the orders share the same ship date.

Scenario 1-B: Multiple orders available at a given time with different ship dates

In this scenario, there may be a trade-off in meeting the delivery dates for the orders. The ESP settings come into effect to keep the delays within acceptable limits.

In a scenario where two or more orders in this set of orders are spaced beyond the maximum number of days specified in the "Allow shipment delay" parameter or the applicable shipping window (the range of dates within which the order must be shipped), only the applicable orders are consolidated together.

For example, if the Ship By dates of orders O1, O2, and O3 are spaced by one day each (D1, D2, and D3), and the allow shipment delay parameter is set to one day, the ESP consolidates either order O1 and O2, or O2 and O3, and not all three. This is because the two-day gap between the Ship By dates of orders O1 and O3 is more than the allowed shipment delay.

Scenario 2: An existing order is held in anticipation of another order to enable consolidation

The one order/shipment available for shipping is held optimistically, in anticipation of another order to the same Ship To Address.

The ESP settings determine how long the order can be held, before it has to be released for further processing. As per the ESP settings, the orders are typically held until they fulfil the weight or volume thresholds, or complete the maximum number of days specified in the "Allow shipment delay" parameter, or the shipments' shipping window.

Shipments' shipping window refers to the window within which a shipment has to be shipped, and is bound by the "Requested Ship Date" on the shipment and the "must ship before date".

The "must ship before date" is computed as follows:

Step 1 Ascertaining customer requested ship date

If a Customer Requested Ship Date exists for the order, it is used.

If a Customer Requested Ship Date is not available for the order, then one of the following is used:

- If the order has a Requested Delivery Date, the Customer Requested Ship Date is computed by subtracting the Global Transit Time from the Requested Delivery Date, and based on the "End of Shift" parameter using the end of shift of the resulting date.
- If Requested Delivery Date is not specified on the order, the current time is assumed as the Customer Requested Ship Date.

Step 2 Calculating Must Ship Before Date

After the Customer Requested Ship Date has been ascertained, the Must Ship Before Date is calculated using the following formula:

```
MustShipBeforeDate = Min (Max (ReqShipDate, CustReqShipDate +
ESPDelayDays), ReqCancelDate))
```

where,

- `MustShipBeforeDate` is the date before which a shipment must be shipped,
- `ReqShipDate` is the earliest ship date as requested by the customer,
- `CustReqShipDate` is the date as ascertained in Step 1.
- `ESPDelayDays` are the number of days the shipment can be delayed from the earliest of the requested ship dates on the orders that make up the shipment. The ESP Delay Days is taken from buyer's inbound compliance or enterprise's outbound constraints or HUB's outbound constraints based on configuration for the freight terms on the order, and based on who maintains ESP parameters.
- `ReqCancelDate` is the last date as specified by the customer for receiving the shipment.

Logic used to consolidate shipments with different Requested Ship Dates

Two shipments with different Requested Ship Dates are consolidated into one shipment only when the Requested Ship Date falls between the Expected Ship Date and the Must Ship Before Date of the shipment to which the new order release is consolidated.

For the shipment consolidation to occur, the Must Ship Before Date should be set in an appropriate way. For instance, if the Requested Cancel Date is less than the Must Ship Before Date, then the Must Ship Before Date should be set to the Request Cancel Date.

Note: Scenario 1-A or 1-B can turn into Scenario 2 when the available multiple orders are consolidated together into one shipment, and this shipment is held in anticipation of another order that may be consolidated into this shipment.

For example, when orders O1 and O2 or O2 and O3 have been consolidated into shipment S1, this shipment may be held in anticipation of another order that may be consolidated into S1.

Although the use of ESP is optional, both buyers and enterprises can establish ESP. When established, a setting in Freight Terms is used to determine which ESP to use first.

13.3.2 Determining Routing

Routing determines how to deliver a shipment. A result of the routing process is that a shipment is assigned to a existing load, or a new load is created for that shipment.

Selling and Fulfillment Foundation can perform routing based on routing guides that are defined within Selling and Fulfillment Foundation, by the use of external routing, or by manual routing. External routing, also known as *dynamic routing,* uses an external resource to determine assignment of a shipment to a load.

13.3.2.1 Routing Guides

Routing Guides are a list of conditions that determine how a shipment should be routed, and what carrier and service should be used. A routing guide has a time period for which is effective, and conditions for when it should be applied. These conditions are based on Freight Terms and Department. Routing guides are maintained within Selling and Fulfillment Foundation.

Each routing guide contains a list of *routing guide lines*, each of which describe detailed conditions for selecting a carrier, service and shipment mode. The conditions include:

- the origin and destination of the shipment
- any carrier service requests
- characteristics of the shipment itself, such as weight and volume.

The routing guide information is based on data used by VICS (Voluntary Inter Industry Commerce Standards) routing.

For example, different departments within the same buyer organization may have separate contracts with various delivery services. Each department could supply a routing guide, which contains its criteria for carrier selection.

An example is the Jolly Toy Company, which has two departments, the Stuffed Toys department, and the Board Games department. The Stuffed Toys department is located in California, and takes all of its deliveries in one location there. The Board Games department works quite differently, it can receive shipments in seven different locations across the US.

The Jolly Toy Company provides a routing guide for each of these departments.

The routing guide for the Stuffed Toys depart is fairly simple; under most circumstances that department uses a single preferred carrier. If the shipment is very heavy, the Stuffed Toys department uses a different carrier.

Table 13–1 Example: Stuffed Toy Department Routing Guide

Conditions	Action
Shipment weight is less than 300 pounds	Ship to San Diego California facility using the carrier service *Happy Toy Carriers*
Shipment weight is more than 300 pounds	Ship to San Diego facility using the carrier service *Big and Bulky Carrier*

The routing guide for the Board Games department is complex; it has a series of conditions based on the origin and destination, weight of the shipment, and requested type of carrier service (next day, ground, and so forth). They are using a range of flexible choices in order to optimize their shipping times and costs.

Table 13–2 Example: Board Game Department Routing Guide

Condition	Action
Shipment weight is less than 5 pound and service is Next Day	Use FedEx
Shipment weight is less than 10 pounds and service is 2nd day	Use UPS
Shipment weight is more than 10 pounds and shipping from CT, NY or RI to NY or NJ	Ship to NJ distributor using the carrier service *NY Special Freight*
Shipment weight is more than 10 pounds and shipping from CT, NY or RI to California, Colorado or Washington	Ship to shipping address, using the carrier service *Cross Country Carrier* or *Big and Bulky Carrier*
and so on.....	

When shipping to the Jolly Toy Company, the enterprise organization consults these Routing Guides. Based on the department to which the shipment is going, the appropriate conditions are assessed, and the appropriate actions are used.

The use of routing guides is optional. Both buyers and enterprises can establish routing guides. If both the buyer and enterprise establish routing guides, a setting in Freight Terms is used to determine which routing guide to use first.

The routing guide for determining a carrier service is complex; it has a series of conditions to be met based on the criteria specified, such as the requested carrier type, cost optimization, and the lowest numbered priority assigned to the carrier.

Table 13–3 Example: Determining Carrier Services

Condition	Action
Shipping Attributes	Ship to Tustin California facility using the carrier service UPSN Ground for shipping hazardous items.
Service charges	For UPSN Ground, the service cost is $200.
Priority	For UPSN Ground, the priority assigned is 2.

13.3.2.2 Dynamic Routing

Dynamic routing uses an external resource to determine how to assign shipments to loads. If dynamic routing is used, then the routing guides are not used.

Dynamic routing can be performed several ways. One approach is to integrate a Transportation Management System (TMS) with Selling and Fulfillment Foundation. For more information about integration, see the *Selling and Fulfillment Foundation: Integration Guide*. A Transportation Management System provides detailed planning of routing of loads.

A second approach to dynamic routing is to implement a User Exit, which then consults an external source for routing planning. For example, the buyer may maintain its own routing system, which takes into account which loads are being routed to which facilities on the day of the proposed load shipment. The buyer then can determine how to route this load. The User Exit can be used to consult the buyer, and update the Selling and Fulfillment Foundation routing processing.

For more information about dynamic routing processing, see Section 13.5.2, "Routing a Shipment".

13.3.2.3 Delivery Plans

A *delivery plan* is a complete sequence of movements needed to deliver one or more orders from one or multiple origins to one or multiple destinations. A delivery plan is comprised of shipments, loads, origins, stops, and destinations. Creating a delivery plan, or manually routing shipments, is done using the Shipment Management Console. A console user determines how a shipment should be routed.

13.4 Inbound Compliance and Outbound Constraints

Both the buyer and the enterprise can have requirements and conditions about how items in an order should be shipped.

13.4.1 Buyers and Inbound Compliance

The term *Inbound compliance* refers to the enterprise conforming to the buyer's conditions about consolidation and delivery.

The buyer can have required conditions which *must* be met when shipping to that buyer. For example, a buyer require orders with different order numbers be shipped in separate shipments. Similarly, a buyer may require orders with different department numbers be shipped in separate shipments.

In the Selling and Fulfillment Foundation, conditions for inbound compliance are established in Participant modeling.

You can establish these conditions for inbound compliance:

Table 13–4 Inbound Compliance Category

Category	Description
Consolidation	Describes if items can be placed in the same shipment. Criteria include: different PO numbers, different order number, different departments, different "Mark For", and different Order Types.
ESP	Whether to use Economic Shipping parameters, and the conditions of those parameters.

Table 13–4 Inbound Compliance Category

Category	Description
Carrier Preferences	A list of preferred carriers. These carrier preferences should be honored, if possible, but a shipment can be made using an alternative carrier, if no matching preferred carrier can be used.
Routing Guide	A guide which describes criteria for how to route deliveries.
Packaging	Describes conditions about mixing items within the same case or pallet. For example, do not mix items with different UOMs in the same pallet. Also provides conditions about when to apply Shipping Container Marking.
Value Added Services	Value Added Services define additional processing or handling used to provide inventory that meets the buyer's requirements. See Chapter 10, "Value-Added Services" for additional information.

13.4.2 Enterprise and Outbound Constraints

The term *Outbound constraints* refers to the own conditions of the enterprise for consolidation and shipping. For example, in order to reduce shipping costs, a enterprise tries to consolidate items by placing as many items as possible into a shipment, until a certain weight is met. The enterprise can set the Economic Shipping Parameters described in Section 13.3.1, "Using Economic Shipping Parameters" to implement this strategy.

In Selling and Fulfillment Foundation, conditions for outbound compliance can be established in the Logistics section of the Distributed Order Management Configuration Console.

You can establish outbound constraints for these categories.

Table 13–5 Outbound Constraints Categories

Category	Description
Consolidation	Describes if items can be placed in the same shipment. Criteria include: different PO numbers, different order number, different departments, different "Mark For", and different Order Types.
Routing Guide	A guide which describes criteria for how to route deliveries.
ESP	Whether to use Economic Shipping parameters, and the conditions of those parameters.

13.4.3 Resolving Conflicting Conditions

You can establish both inbound compliance (the buyer's conditions) and outbound constraints (the conditions of the enterprise). If both are established, there may be conflicting conditions when determining shipment consolidation or routing.

For example, both the buyer and the enterprise have established Economic Shipping Parameters. In this case, the buyer is willing to wait longer than the enterprise in order consolidate items into a shipment.

Table 13–6 Buyer and Enterprise Economic Shipping Parameters

ESP Attribute	Settings	
	Buyer	Seller
Weight threshold	300 kilos	150 kilos
Volume threshold	40 cubic meter	20 cubic meter
Days to hold	7	4

To resolve this conflict, a simple setting is used determine which order to apply the preferences. When establishing the Freight Terms in Distributed Order Management, you can establish the order, which can be:

- First Buyer then Enterprise
- First Enterprise then Buyer

For example, if *First Buyer then Enterprise* is selected, then the Buyer's settings are used. Shipments are consolidated until one of the following occurs:

- weight threshold of 300 kilos is met
- volume threshold of 40 cubic meters is met
- 7 days have elapsed.

13.5 Outbound Shipments

The process for creating shipment starts when order lines are released for shipment, which creates an order release, and completes when the shipment is delivered. This section addresses:

- Creating a Shipment
- Routing a Shipment
- Outbound Shipment Pipeline
- Outbound Shipment Console

13.5.1 Creating a Shipment

When order lines are released for shipment, creating an order release. The following figure illustrate the logic used. The details of this logic can be changed by customizing the Outbound Shipment Pipeline.

Figure 13–2 Create Shipment Logic

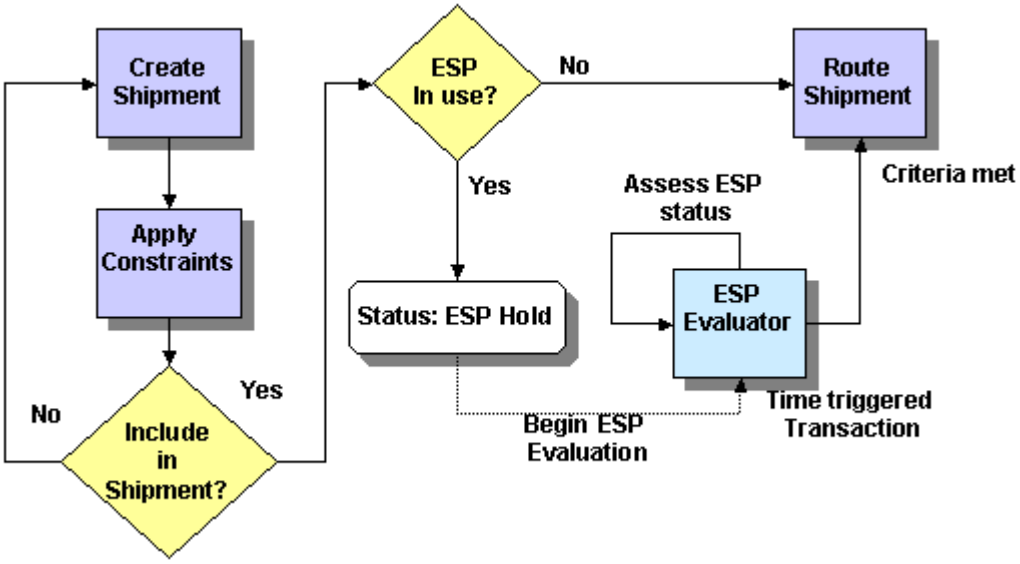


Figure 13–2, "Create Shipment Logic" shows a process for assigning order release items to a shipment.

13.5.1.1 Constraints and Creating an Order Release

When an order release is created, it is created by applying a series of rules to ensure that the items in an order release can be shipped together. These rules include:

- Mandatory do-not-mix constraints - These constraints are part of the Selling and Fulfillment Foundation system. A partial list of items that cannot be shipped together is items with different: Ship Node, Ship To Address, Freight Terms, or Buyer Organization. The complete list is defined in the Sales Order document type.
- Inbound compliance - Optional buyer based rules, such as do not ship items together which have different: Customer PO #, Order Number, Department, or Mark For.

- Outbound constraints - Optional enterprise based rules, such as do not ship items together which have different: Customer PO #, Order Number, Department, or Mark For.
- Routing based constraints - Based on the routing guide, if one is in use.

By assessing these constraints, the system creates order releases that can be shipped together. This assessment is validated by the create shipment processing.

13.5.1.2 Consolidating shipments

In Selling and Fulfillment Foundation, orders released from Sterling Commerce or third-party ERPs, or both can be consolidated to shipments. Consolidation of Shipments is carried out based on various parameters, including Ship To, Buyer, Seller, Mark For, and Economic Shipping Parameters (ESP).

For example, a buyer may frequently place orders that must be split into several shipments, based on some condition, such as order lines being associated with different departments. The items going to the *same* department may be consolidated, if ESP processing is used. Here, the routing process begins after the ESP conditions are applied and met.

13.5.2 Routing a Shipment

Routing refers to the assignment of the optimal carrier and mode to ship a shipment. A mode refers to parcel, less than truck load (LTL) and truck load (TL). The choices available when routing for a shipment is required, include the requirement to consider node or business entity (enterprise) or customer specific criteria when determining the carrier. A carrier and mode of shipment is chosen after using the appropriate guide as per the criteria and also factoring in the destination and weight of the shipment. For example, Parcel carriers often have weight limits or do not supply to specific zip codes (postal codes).

Sterling WMS supports the ability to hold a routing guide at the node, enterprise or customer level. The routing guide also captures the effective dates for the guide. It is typical in an industry that the routing guides are changed by a buyer every two or three years.

The ability to integrate with an external routing guide or a Transportation Management System is also available.

A shipment in Sterling WMS can have a pre-assigned carrier and service, in which case the routing guide is not used. A shipment can specify the service requested. For example, Best Parcel, which in turn is used in the routing guide to optimize the carrier and shipment mode chosen.

The outbound shipment pipeline defines the requirement for a shipment to be routed. The following figure illustrates the logic used.

Figure 13–3 Routing Processing Overview

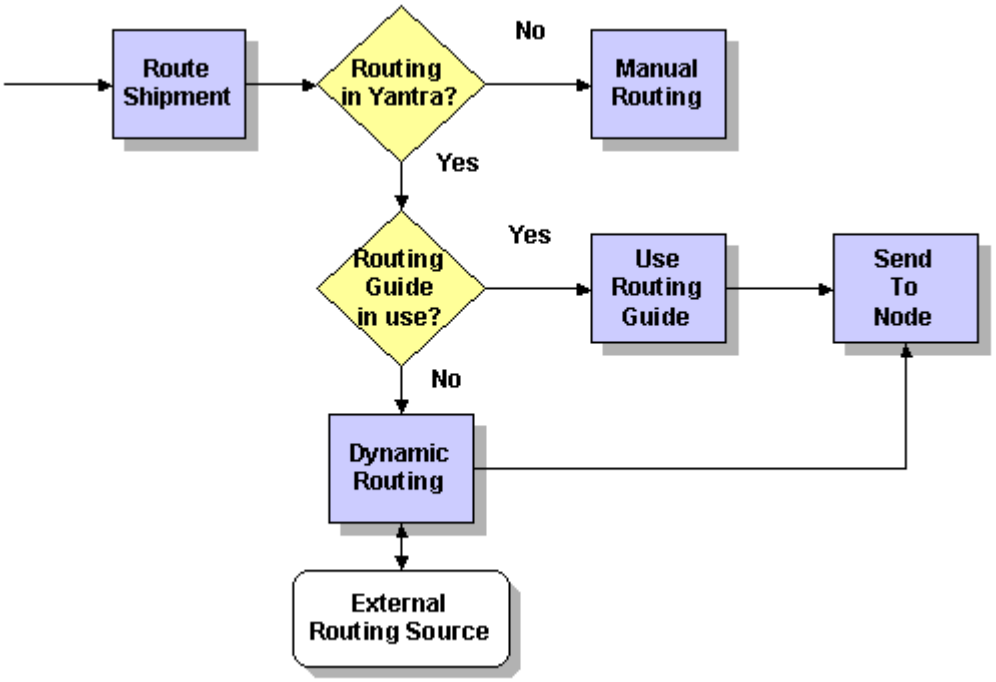


Figure 13–3, "Routing Processing Overview" shows an overview of routing. There are three ways that routing can be performed:

- Use Routing Guide - Routing is based on the Buyer or Enterprise routing guides
- Dynamic - Routing is based on an external resource, such as a Transport Management System or a customized interaction with the buyer.

- Manual Routing - Shipment routing is established by the console operator.

13.5.2.1 Using Routing Guides

A routing guide in Sterling WMS allows specifying optimal carrier and mode of shipment (parcel, LTL or TL) based on criteria including requested service, weight, zip codes (postal codes). In situations where multiple routing guides are used, the system can be configured to consider the Buyer's routing guide first to determine a carrier. If the Buyer's routing guide fails to determine a carrier, then the system considers the Enterprise's routing guide. If this fails, then the system considers the Hub's routing guide.

The system can also be configured to consider the Seller's routing guide first to determine a carrier. If this fails, the system considers the Hub's routing guide. However, in this case, the Buyer's routing guide is not considered.

If a parcel mode of shipment is chosen, then the shipment is marked as routed. Further processing is not required for this shipment. *It is possible to consolidate new orders into a shipment, in which case re-routing is done for the shipment.*

For LTL shipments, shipments are added to an existing load based on the criteria or a new load is created. A load is grouping of shipments that is shipped to the same delivery address. All LTL shipments are automatically converted to TL shipments when the shipment volume exceeds the volume specified for a TL. Each time a shipment is added to an existing LTL load, re-routing is done for the new load. The shipment is marked as routed. For shipments that are routed as TL, there is no further consolidation attempted. The shipment is marked as routed.

13.5.3 Zone Skipping

Typically, carriers group several geographical regions into zones and provide shipping charges for movement of shipments between these zones. Carriers have consolidation and break bulk centers in each zone from where the local shipments within the zone are delivered.

When shipping shipments between zones, warehouses may find it profitable to ship a shipment as an LTL or TL to a carrier in the destination zone. These shipments are later shipped within the zone as

parcel shipments. Due to this, warehouses skip several zones that are between the source and destination zones.

The Zone Skipping functionality of Sterling WMS enables a warehouse to reduce transportation costs by consolidating individual shipments into an LTL or TL load, shipping them to a region and taking it to the carrier's break bulk node for that region. This consolidated load is called a Break Bulk Load. Upon reaching the break bulk node, the break bulk load is dismantled and individual shipments are shipped to their destinations based on their SCAC and carrier service.

13.5.3.1 Routing in Zone Skipping

During zone skipping, shipments are routed as follows:

- From ship node to the break bulk node
- From the break bulk node to the destination node

The SCAC and carrier service of the shipment is set up according to the routing guide. The system determines if a break bulk load exists from ship node to the break bulk node. If a break bulk load is present, the shipment is added to it. If no such load is present, the system checks for other shipments that satisfy the following conditions:

- The routing source is Sterling WMS.
- The routing guide is maintained in Sterling WMS.
- The carrier type of the shipment is parcel.
- The shipment is not manifested.
- Freight charges are paid by the shipper.

If shipments matching the above criteria are found, the system prepares a consolidated list of all shipments and determines if this consolidated shipment satisfies the minimum weight or volume criteria specified for that break bulk node. The system creates a load and adds these shipments to the load. The origin of this load is marked as the ship node and the destination as break bulk node. The load is routed and marked as a break bulk load.

Note: Any changes made to a shipment that belongs to the break bulk load is propagated to the load. If a break bulk load is deleted, the shipments belonging to it are automatically re-routed.

A break bulk load is deleted, if the following conditions are met:

- None of the shipments belonging to the load are manifested.
- If the minimum weight and volume criteria are not met.

13.5.3.2 Manifesting for Shipments' Shipping Zones

As a part of routing, the Sterling Warehouse Management System considers parcel shipments that satisfy the following conditions for zone skipping:

- The shipment is stamped with a break bulk node.
- The shipment is not a part of any manifest.
- The shipment maintains a routing guide in Sterling WMS.
- The routing source is assigned by Sterling WMS.

On meeting the zone skipping threshold weight and volume, the system creates an LTL or TL load to carry these shipments to the break bulk node. Shipments that belong to a break bulk load are automatically manifested from the break bulk node.

Note: You must configure a break bulk node as a non-WMS ship node. You must also configure a different break bulk node for each ship node to ensure that all shipments belonging to the manifest of a break bulk node arrive at the break bulk node at the same time (as part of break bulk load). The close manifest process is not dependent on the arrival of loads of other ship nodes.

The system automatically manifests all shipments belonging to a break bulk load in the following ways:

- Shipment Level Integration—All containers of a shipment are manifested when the shipment pack process is complete.
- Package Level Integration—Each container of a shipment is manifested when its pack process is complete.

If the user unpacks a manifested container, the shipments are automatically un-manifested. For shipments that have shipment level integration with FedEx, all containers of the shipment are un-manifested.

The open manifest at the break bulk node is closed when you confirm a load. Shipments that belong to a break bulk load are confirmed when the load is shipped from the origin node.

13.5.3.3 Determining the Second Leg Carrier for Parcel Shipments

The Sterling Warehouse Management System provides the capability to route shipments (that are a part of the break bulk load) from the break bulk node to the final destination. To achieve this, the user needs to configure a routing guideline from the break bulk node to the final destination.

If the second leg routing fails for parcel shipments, then for:

- Domestic Shipments—The original SCAC and carrier service code is stamped on the shipment.
- International Shipment—The "Could not determine SCAC for second leg" routing error code is stamped on the shipment. Despite this, the shipment goes to the "Shipment Routed" status. You must manually resolve the error.

13.5.4 Outbound Shipment Pipeline

The actual processing of a outbound shipment flows through a set of transactions and statuses until its completion. This chain of transactions and shipment statuses is called the outbound shipment pipeline.

The outbound shipment pipeline provides Selling and Fulfillment Foundation with a means to perform actions, such as sending notifications or logging alerts, dependent on the location of the outbound

shipment in its pipeline. It also provides you with a means to track an outbound shipment from creation to delivery.

The outbound shipment pipeline from which your business runs is unique to how your system administrator has set up your business environment. However, every outbound shipment pipeline generally begins with a transaction that creates an outbound shipment and ends with a transaction that indicates an order has been delivered.

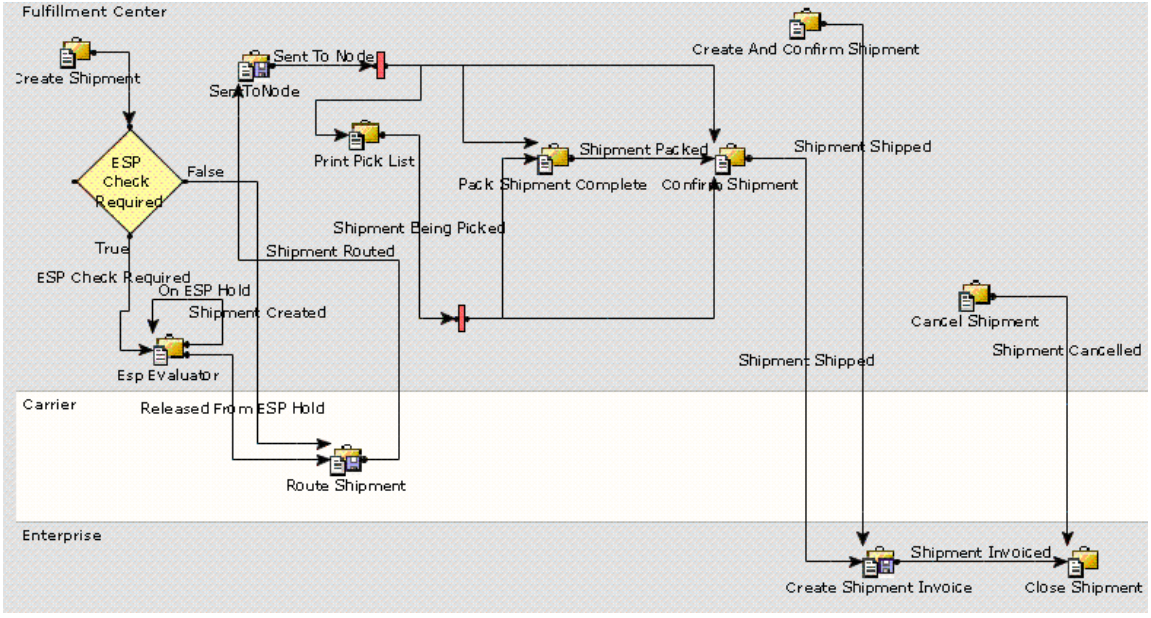
A transaction is an occurrence that needs to be tracked and can call certain actions to be performed. A shipment status describes what state an outbound shipment is in and moves it from transaction to transaction.

The statuses that an outbound shipment can flow through in the default outbound shipment pipeline of Selling and Fulfillment Foundation are:

- Shipment Created - The shipment has been created from an order release or a chained order.
- Shipment Being Picked - The line items are physically being picked in preparation for shipment.
- Shipment Shipped - The shipment has been shipped to the ship to address.
- Shipment Delivered - The shipment has been delivered to the ship node address.
- Included In Receipt - The shipment has been included in the receipt.
- Receipt Closed - The shipment has been received and is considered complete.
- Shipment Invoiced - An invoice has been created for the shipment.
- ESP Check Required - The ESP evaluator must be run to determine if ESP conditions have been met.
- On ESP Hold - The shipment is being held until ESP conditions are met.
- Released From ESP Hold - The ESP conditions have been met, and the ESP Hold is lifted.
- Shipment Routed - The shipment has been assigned to a load.
- Sent To Node - The shipment has been sent to be packed.
- Shipment Cancelled - The shipment has been cancelled

The following figure is a graphical representation of the default outbound shipment pipeline as it appears in the Applications Manager.

Figure 13–4 Outbound Shipment Pipeline



13.5.5 Outbound Shipment Console

The Outbound Shipment Console provides information on outbound shipments made from sales orders and shipment containers. It provides Sellers, node users, and Carriers with information necessary to ensure on-time shipment of the correct orders.

An order becomes an outbound shipment when an order release is packed and it is physically ready to be shipped by the shipping node. A shipment details the ship to address, dimensions, shipment charges, carrier information, and information about the lines shipped.

A outbound shipment as an entity can be used in a delivery plan by attaching it to a load. For more information about setting up delivery plans and loads, see the *Sterling Logistics Management: User Guide*.

13.6 Bundle Shipments

13.6.1 Creating a Bundle Shipment

When order information for a bundle is available, the creation of a shipment for a bundle occurs based on whether a bundle parent is added to a shipment or the components of a bundle are added to a shipment.

When a bundle parent is added to a shipment the components of the bundle are not automatically added to the shipment.

When a bundle component is added to a shipment or a shipment is created for a component, its parent line and all n-level lines are also added to the shipment if they have not been added previously.

When order information for a bundle is not available, the components of the bundle are taken from the catalog.

13.6.2 Confirming a Bundle Shipment

When shipping the last component in a bundle, the status of a parent line changes to shipped or its extended status, based on the pipeline setup.

When the ratio of a bundle component is not maintained, the order line status is only updated when the entire bundle ships.

For a bundle parent, confirm shipment quantity is stamped as the max of the component quantity at Shipment level. However, on the OrderLine the exact parent quantity shipped is shown.

13.6.3 Invoicing a Bundle Shipment

The Enterprise level rule "When to invoice Bundle Parent Line" determines how a bundle is invoiced. This rule has two values: First and

Last. When the rule is set to First, a bundle is invoiced whenever one of its components are shipped. When the rule is set to Last, a bundle is invoiced only when all of its components are shipped.

Example 13–1 explains how a bundle is invoiced based on this Enterprise level rule.

Example 13–1 Customer Orders Two Bed Set Bundles

In this example, a customer has ordered two Bed Sets. A Bed Set is a bundle that consists of the following:

- Bed Set (Parent Bundle)
  - Mattress
  - Bed Frame

Table 13–7 demonstrates how a bundle is invoiced if the Enterprise level rule "When to invoice Bundle Parent Line" is set to the values First or Last.

Table 13–7 Invoicing a Bundle Shipment

Date Shipped	What is Shipped	Amount Invoiced to the Parent Bundle	
		First	Last
08/01	1 Bed Set	1	1
08/05	2 Mattresses	2	0
08/10	2 Bed Frame	0	2
08/20	2 Bed Sets	2	2

13.7 Loads

A *load* carries one or more complete shipments (never a partial shipment) between two points. Loads are created as a result of the routing for a shipment. See Section 13.5.2, "Routing a Shipment" for more details.

This section discusses:

- Creating a Load
- Load Execution Pipeline

13.7.1 Creating a Load

Creating a load is done as a part of the routing process. During routing processing, shipments can either be added to existing loads, or a new load is created. A new load may be created either because no load is currently being shipped to the location where the shipment is going, or because current loads cannot accommodate the shipment.

For more information, see Section 13.5.2, "Routing a Shipment".

13.7.2 Load Execution Pipeline

Beginning from its creation, a load document flows through a set of transactions and statuses until its completion. This chain of transactions and shipment statuses is called the *load execution pipeline*.

The load execution pipeline provides Selling and Fulfillment Foundation with a means to perform actions, such as sending notifications or logging alerts, dependent on the location of the load in its pipeline. It also provides you with a means to track a load from creation to delivery.

The load execution pipeline from which your business runs is unique to how your system administrator has set up your business environment. However, every load execution pipeline generally begins with a transaction that creates a load and ends with a transaction that indicates a load has reached its assigned destination stop.

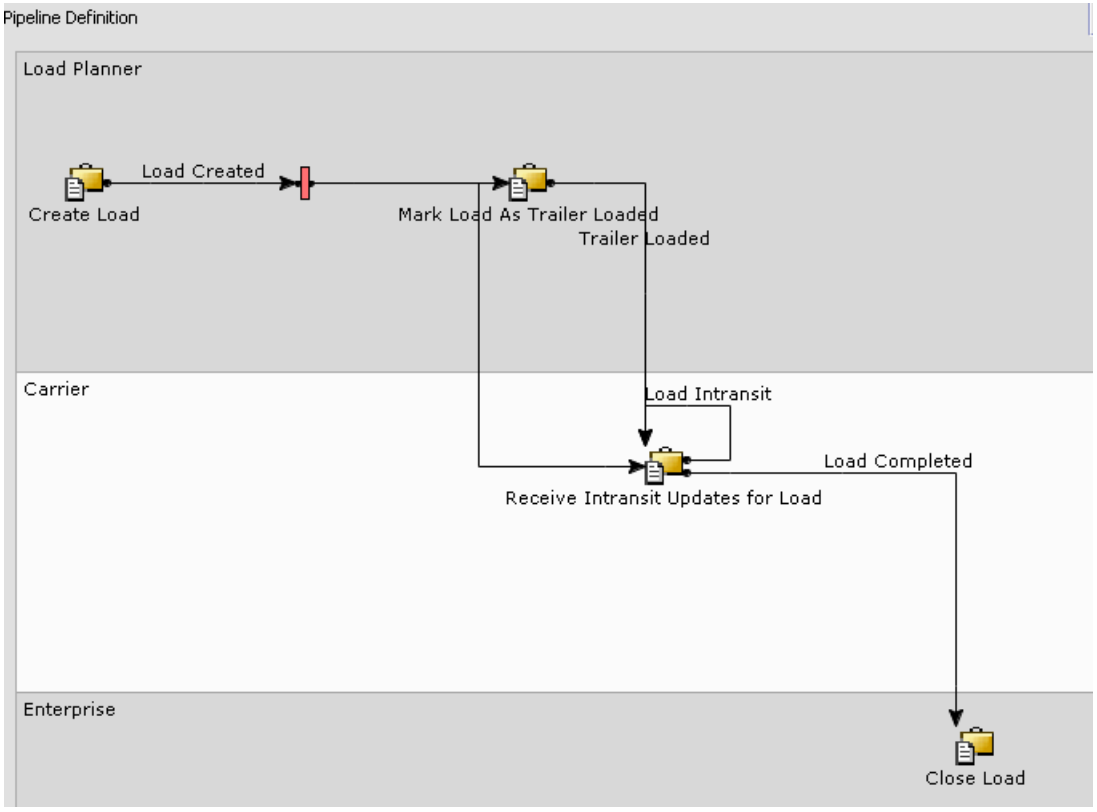
A transaction is an occurrence that needs to be tracked and can call certain actions to be performed. A load status describes what state a load is in and moves it from transaction to transaction.

The statuses that a load can flow through in the default load execution pipeline of Selling and Fulfillment Foundation are:

- Load Created
- Trailer Loaded
- Load In Transit
- Load Closed

The following figure is a graphical representation of the default load execution pipeline as it appears in the Applications Manager.

Figure 13–5 Load Execution Pipeline



13.8 Inbound shipments

An order becomes an inbound shipment when a purchase order release is packed and it is physically ready to be shipped by the shipping node.

This section describes:

- Inbound Shipment Pipeline
- Inbound Shipment Console

13.8.1 Inbound Shipment Pipeline

Beginning from its creation, an inbound shipment flows through a set of transactions and statuses until its completion. This chain of transactions and inbound shipment statuses is called the inbound shipment pipeline.

The inbound shipment pipeline provides Selling and Fulfillment Foundation with a means to perform actions, such as sending notifications or logging alerts, dependent on the location of the inbound shipment in its pipeline. It also provides you with a means to track an inbound shipment from creation to delivery.

The inbound shipment pipeline from which your business runs is unique to how your system administrator has set up your business environment. However, every inbound shipment pipeline generally begins with a transaction that creates an inbound shipment and ends with a transaction that indicates an order has been delivered.

A transaction is an occurrence that needs to be tracked and can call certain actions to be performed. A inbound shipment status describes what state an inbound shipment is in and moves it from transaction to transaction.

The statuses that an inbound shipment can flow through in the default inbound shipment pipeline of Selling and Fulfillment Foundation are:

- Shipment Created - The shipment has been created from an order release or a chained order.
- Shipment Being Picked - The line items are physically being picked in preparation for shipment.
- Shipment Shipped - The shipment has been shipped to the ship to address.
- Shipment Delivered - The shipment has been delivered to the ship node address.
- Included In Receipt - The shipment has been included in the receipt.
- Shipment Invoiced - An invoice has been created for the shipment.

13.8.2 Inbound Shipment Console

The Inbound Shipment Console provides information on inbound shipments made from purchase orders and inbound shipment containers.

It provides Sellers, node users, and Carriers with information necessary to ensure on-time inbound shipment of the correct orders.

An order becomes an inbound shipment when a purchase order release is packed and it is physically ready to be shipped by the shipping node. An inbound shipment details the ship to address, dimensions, inbound shipment charges, carrier information, and information about the lines shipped.

An inbound shipment as an entity can be used in a delivery plan by attaching it to a load. For more information about configuring delivery plans and loads, see the *Sterling Logistics Management: Configuration Guide*.

13.9 Hold Processing

Selling and Fulfillment Foundation allows a shipment or load to be held for multiple reasons. Holds of a certain type prevent a shipment or load from being processed by transactions that are associated with that hold type. Holds of certain types can also prevent certain modifications to be applied to a shipment or load. If an enterprise wants to perform an address verification every time an address is changed or a date verification every time a delivery date is changed, the shipment or load needs to be kept from being processed until those checks are completed.

A hold can be in three different statuses:

- `Created`, when the hold has just been applied to a shipment or load, and has not yet been examined.
- `Rejected`, when the hold has been reviewed by a supervisor who decided that the shipment or load should not be processed.
- `Resolved`, when the hold has been reviewed by a supervisor who decided that the shipment or load should be processed.

Hold types can be applied in the following ways:

- Manually: a supervisor may feel that a shipment or load needs to be placed on hold for fraud check, and applies the hold through the Application Console. It is possible to configure a particular hold type so that only users of a specific group, or set of groups, can apply the hold to a shipment or load.

- Automatically on shipment or load creation: every time a shipment or load is created, it is placed on a specific hold type by default.
- Automatically upon resolution of another hold type: if a certain hold type is resolved, it can trigger another hold type automatically. This is specified in the hold type that is being applied on resolution of the other.
- Automatically when a specific modification type occurs: hold types can be configured to be placed automatically when a certain modification type occurs at the shipment or load level, for instance a terrorist threat hold when there is a tip-off that a load may contain bombs or drugs for a terrorist organization.

Independent of how the hold is applied, it is possible to specify a condition that determines whether or not the shipment or load should be put on hold. For example, you could want to place only shipments with a specific delivery address on hold. You can use the condition builder to do this on a hold type. For more information about using the condition builder, see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide*.

By default, all transactions are allowed to process a shipment or load that is on hold. Using the Applications Manager, you can specify which transactions are prevented from processing a shipment or load that is in a particular hold type. Transactions that can be configured to not process shipments or loads on hold are known as hold type enabled transactions, and are defined at the document type level for a given enterprise. Custom transactions that are not derived from an abstract transaction are all hold type enabled.

Two transactions need to be configured to process shipments or loads that are in a hold type, and then remove them from that hold: one for holds in `created` status, and another for holds in `rejected` status. Additionally, holds can be resolved manually, and it is possible to specify a set of user groups with the authorization to do so. That, too, is done at the document type level for a given enterprise, in the Applications Manager.

14

Reverse Logistics

Reverse Logistics involves managing the return process for your products. Users can view return order information to inspect returned items, determine how returned items should be handled, and provide the Buyer with the necessary amount of credit for the return.

14.1 Parts of a Return Order

In Selling and Fulfillment Foundation, a return order can be broken down into three levels, the return order header level, return order line level, and return order release level. The return order header level contains all of the return order lines that have been returned, the return order line level is broken down by each individual line that has been returned, and the return order release level contains all of the lines that have been released to a return node.

14.2 Reverse Logistics Pipeline

Beginning from its creation, a return order flows through a set of transactions and statuses until its completion. This chain of transactions and return order statuses is called the reverse logistics pipeline. There are two default pipelines, the standard Reverse Logistics pipeline and the Consumer Returns pipeline used to handle customer returns to the retail site.

A reverse logistics pipeline provides Selling and Fulfillment Foundation with a means to perform actions, such as sending notifications or logging alerts, dependent on the location of the return order in the pipeline. It also provides you with a means to track a return order from creation to completion and perform any manual interventions, if necessary.

The reverse logistics pipeline from which your business runs is unique to how your system administrator has set up your business environment. However, every reverse logistics pipeline generally begins with a transaction that creates a return order and ends with a transaction that indicates a return order has been closed.

A transaction is an occurrence that needs to be tracked and can call certain actions to be performed. A return order status describes what state a return order is in and moves it from transaction to transaction.

The following statuses may be used in a Reverse Logistics pipeline, depending on how it is configured within your system:

- Authorized - The return has been authorized by the applicable parties as per your business practices.
- Awaiting Exchange Order Creation - The order has been returned and an exchange order needs to be created.
- Cancelled - The return has been cancelled.
- Created - The return has been created.
- Draft Created - A draft return has been created for a blind return in the Create Return Console. All aspects of this return can be modified until it is confirmed.
- Exchange Order Created - An exchange order has been created.
- Held - For any reason, the purchase order is being held and no modifications can be made until it is released from the hold.
- Included In Shipment - The return is included in a shipment.
- Inspected - The return has been inspected by the return node.
- Inspected As Components - One or more individual components have been inspected.
- Not Authorized - The return has not yet been authorized by the applicable parties as per your business practices.
- Not Released - The return has not been released to the return node.
- Received - The return has been received by the return node.
- Received As Components - The return has been received as one or more individual components.

- Receipt Closed - The necessary return handling has been completed and the return is closed.
- Released - The return order document has been released to the applicable return node.
- Removed From Release - One or more items in the return have been removed from the release.
- Return Invoiced - An invoice has been created for the return.
- Shipped - The return has been shipped.
- Shorted - The return contains less quantity than originally ordered. The return is closed.
- Unreceived - The return has not been received by the return node.

Figure 14–1, Figure 14–2, Figure 14–3, and Figure 14–4 illustrate the default pipelines displayed in the Applications Manager:

- Reverse Logistics Pipeline
- Consumer Returns Pipeline
- Reverse Logistics Service Pipeline
- Reverse Logistics Pipeline For Bundle Parent

Figure 14–1 Reverse Logistics Pipeline

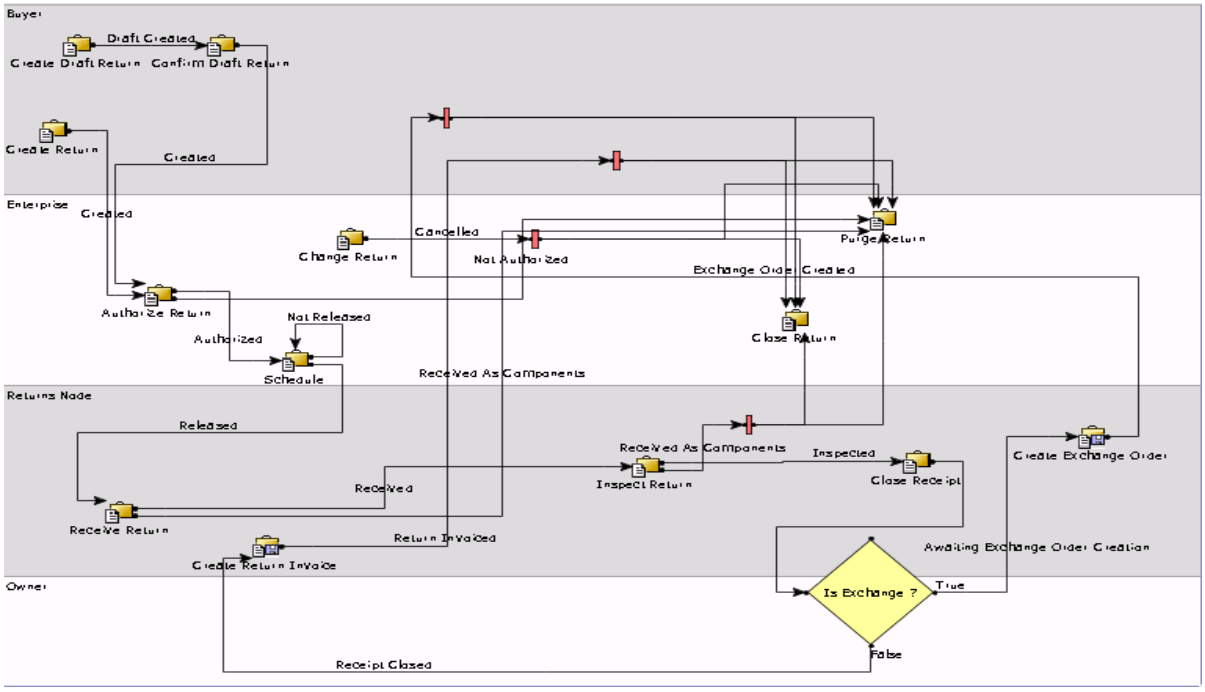


Figure 14–2 Consumer Returns Pipeline

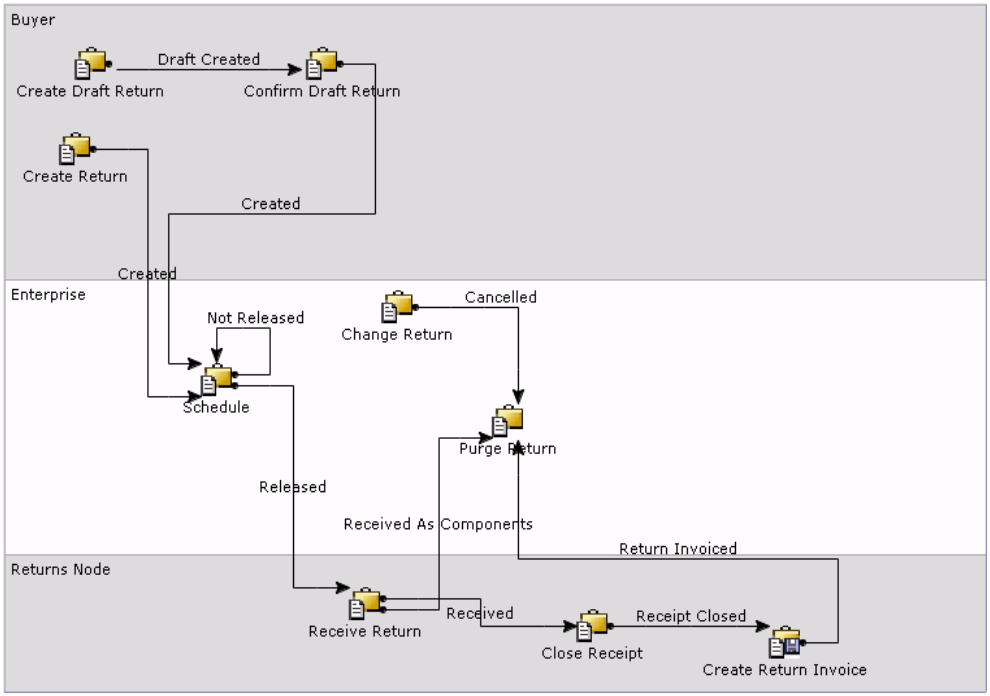


Figure 14–3 Reverse Logistics Service Pipeline

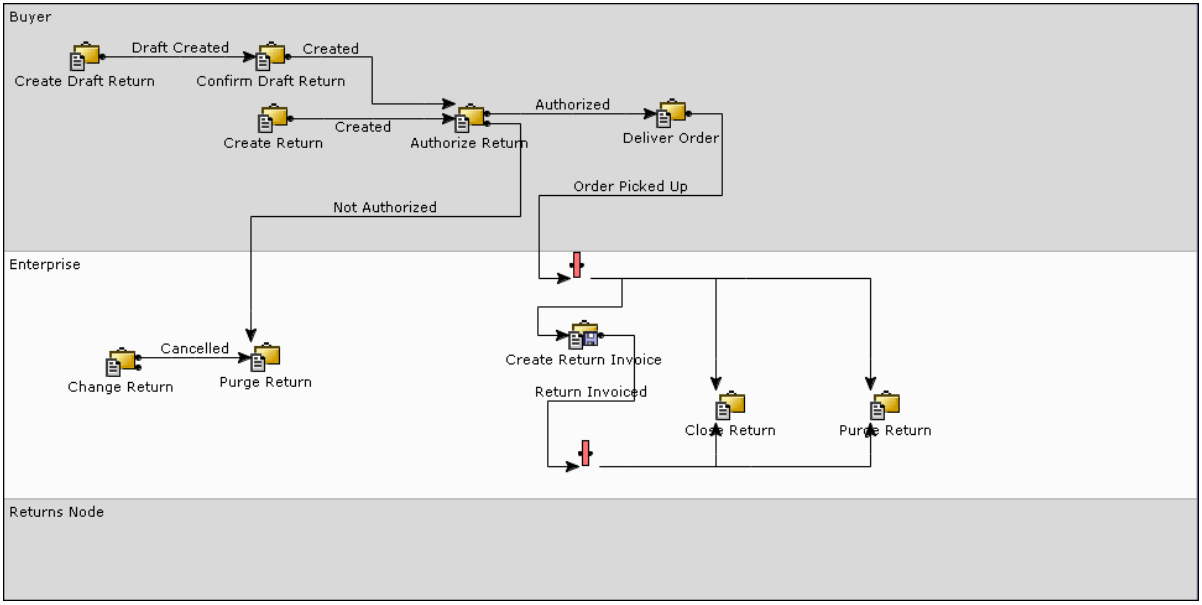
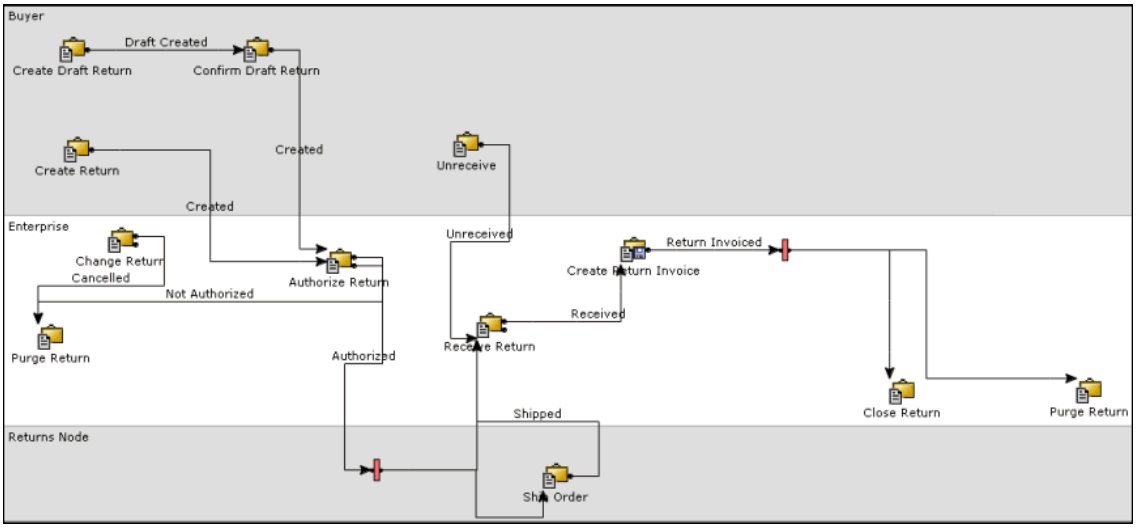


Figure 14–4 Reverse Logistics Pipeline For Bundle Parent



Index

A

B

C

D

E

F

G

H

I

K

L

M

N

O

P

Q

R

S

T

U

V

W