Sterling Commerce
An IBM Company

Selling and Fulfillment Foundation: Extending Transactions Guide

Release 8.5

October 2009

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. Copyright (c) 2008 Azer Koçulu http://azer.kodfabrik.com. Copyright © Einar Lielmanis, einars@gmail.com. Copyright (c) 2006 John Reilly (www.inconspicuous.org) and Copyright (c) 2002 Douglas Crockford (www.crockford.com). Copyright (c) 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 (C) Microsoft Corp. 1981-1998. Purple Technology, Inc. Copyright (c) 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 (c) 2001-2003 Ant-Contrib project. Copyright © 1998-2007 Bela Ban. Copyright © 2005 Eclipse Foundation. Copyright © 2002-2006 Julian Hyde and others. 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 Raghu K, 2003. Copyright © 2004 David Schweinsberg. Copyright © 2005-2006 Darren L. Spurgeon. Copyright © S.E. Morris (FISH) 2003-04. Copyright © 2006 VisualSoft Technologies. Copyright © 2002-2009 Zipwise Software. All rights reserved by all listed parties.

Certain components of the Sterling Commerce Software are distributed on the same storage media as Third Party Software which are not listed above. Additional information for such Third Party Software components of the Sterling Commerce Software is located at: installdir/mesa/studio/plugins/SCI_Studio_License.txt.

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 (c) 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/8.5/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/mesa/gisdav/WEB-INF/lib/Slide_License.txt,
<install>/mesa/studio/plugins/xerces_2.7_license.txt,
<install>/jar/commons_pool/1_2/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 (c) 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 (c) 1999, IBM Corporation., http://www.ibm.com.

- software copyright (c) 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 (c) 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 (c) 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

DOM4J Software

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>/SI/repository/rcp/rcpdependencies/windows/eclipse/notice.html,
<install_dir>/SI/repository/rcp/rcpdependencies/windows/eclipse/plugins/notice.html,
<install_dir>/SI/repository/rcp/rcpdependencies/gtk.linux_x86/eclipse/notice.html, and
<install_dir>/SI/repository/rcp/rcpdependencies/gtk.linux_x86/eclipse/plugins/notice.html.

A copy of the EPL is found at
<install_dir>/SI/repository/rcp/rcpdependencies/windows/eclipse/plugins/epl-v10.htm,
<install_dir>/SI/repository/rcp/rcpdependencies/windows/eclipse/epl-v10.htm,
<install_dir>/SI/repository/rcp/rcpdependencies/gtk.linux.x86/eclipse/plugins/epl-v10.html, and
<install_dir>/SI/repository/rcp/rcpdependencies/gtk.linux.x86/eclipse/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 ehache v.1.5 software (Copyright © 2003-2008 Luck Consulting Pty. Ltd.) ("Ehache 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/8.5/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."

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:

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.

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.

JSLib Software

The Sterling Commerce Software is distributed with or on the same storage media as the JSLib software product (Copyright (c) 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/MPL-1.1.html.

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."

Purple Technology

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

Copyright (c) 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 (c) 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>/3rdParty/rico license.doc. 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.

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 - <install>/Applications/Foundation/lib

SUN ACTIVATION JAR - <install>/ Applications/Foundation/lib

SUN JavaMail - <install>/jar/javamail/1_4/LICENSE.txt

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 the CDDL is found in http://kenai.com/projects/javamail/sources/mercurial/show.

The source code for the Web-App Software may be found at: <install>/3rdParty/sun/javamail-1.3.2/docs/JavaMail-1.2.pdf

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

1

Checklist for Customization Projects

This chapter provides a high-level checklist for the tasks involved in customizing or extending Selling and Fulfillment Foundation.

1.1 Customization Projects

Projects to customize or extend Selling and Fulfillment Foundation vary with the type of changes that are needed. However, most projects involve an interconnected series of changes that are best carried out in a particular order. The checklist identifies the most common order of customization tasks and indicates which guide in the documentation set provides details about each stage.

1. Prepare your development environment

   Set up a development environment that mirrors your production environment, including whether you deploy Selling and Fulfillment Foundation on a WebLogic, WebSphere, or JBoss application server. Doing so ensure that you can test your extensions in a real-time environment.

   You install and deploy Selling and Fulfillment Foundation in your development environment following the same steps that you used to install and deploy Selling and Fulfillment Foundation in your production environment. Refer to Selling and Fulfillment Foundation system requirements and installation documentation for details.

   An option is to customize Selling and Fulfillment Foundation with Microsoft COM+. Using COM+ provides you with advantages such as increased security, better performance, increased manageability of server applications, and support for clients of mixed environments. If

this is your choice, see the *Selling and Fulfillment Foundation: Customization Basics Guide* about additional installation instructions.

2. Plan your customizations

   Are you adding a new menu entry, customizing the Sign In screen and logo, creating new themes, customizing views and wizards, or adding new screens? Each type of customization varies in scope and complexity. For background, see the *Selling and Fulfillment Foundation: Customization Basics Guide*, which summarizes the types of changes that you can make.

   Important guidelines about file names, keywords, and other conventions are found in the *Selling and Fulfillment Foundation: Customization Basics Guide*.

3. Extend the Database

   For many customization projects, the first task is to extend the database so that it supports the other UI or API changes that you make later. For instructions, see the *Selling and Fulfillment Foundation: Extending the Database Guide* which include information about the following topics:

   - Important guidelines about what you can and cannot change in the database.
   - Information about modifying APIs. If you modify database tables so that any APIs are impacted, you must extend the templates of those APIs or you cannot store or retrieve data from the database. This step is required if table modifications impact an API.
   - How to generate audit references so that you improve record management by tracking records at the entity level. This step is optional.

4. Make other changes to APIs

   Selling and Fulfillment Foundation can call or invoke standard APIs or custom APIs. For background about APIs and the services architecture in Selling and Fulfillment Foundation, including service types, behavior, and security, see the *Selling and Fulfillment Foundation: Customizing APIs Guide*. This guide includes information about the following types of changes:

- How to invoke standard APIs for displaying data in the UI and also how to .save the changes made to the UI in the database.
- Invoke customized APIs for executing your custom logic in the extended service definitions and pipeline configurations.
- APIs use input and output XML to store and retrieve data from the database. If you don't extend these API input and output XML files, you may not get the results you want in the UI when your business logic is executing.
- Every API input and output XML file has a DTD and XSD associated to it. Whenever you modify input and output XML, you must generate the corresponding DTD and XSD to ensure data integrity. If you don't generate the DTD and XSD for extended Application XMLs, you may get inconsistent data.

5. Customize the UI

   Sterling Commerce applications support several UI frameworks. Depending on your application and the customizations you want to make, you may work in only one or in several of these frameworks. Each framework has its own process for customizing components like menu items, logos, themes, and etc. Depending on the framework you want, consult one of the following guides:

   - *Selling and Fulfillment Foundation: Customizing Console JSP Interface for End User Guide*
   - *Selling and Fulfillment Foundation: Customizing the Swing Interface Guide*
   - *Selling and Fulfillment Foundation: Customizing User Interfaces for Mobile Devices Guide*
   - *Selling and Fulfillment Foundation: Customizing the RCP Interface Guide* and *Selling and Fulfillment Foundation: Using the Sterling RCP Extensibility Tool Guide*
   - *Customizing the Web UI Framework Guide*

6. Extend Transactions

   You can extend the standard Selling and Fulfillment Foundation to enhance the functionality of your implementation of Selling and Fulfillment Foundation and to integrate with external systems. For background about transaction types, security, dynamic variables, and extending the

Condition Builder, see the *Selling and Fulfillment Foundation: Extending Transactions Guide Selling and Fulfillment Foundation: Extending the Condition Builder Guide* . These guides includes information about the following types of changes:

- How to extend Selling and Fulfillment Foundation Condition Builder to define complex and dynamic conditions for executing your custom business logic and using a static set of attributes.
- How to define variables to dynamically configure properties belonging to actions, agents, and services configurations.
- How to set up transactional data security for controlling who has access to what data, how much they can see, and what they can do with it.
- How to create custom time-triggered transactions. You can invoke and schedule these custom time-triggered transactions in much the same manner as you invoke and schedule Selling and Fulfillment Foundation standard time-triggered transactions. Finally, you can coordinate your custom, time-triggered transactions with external transactions and run them either by raising an event, calling a user exit, or invoking a custom API or service.

7. Build and deploy your customizations or extensions

   After performing the customizations that you want, you must build and deploy your customizations or extensions. First, build and deploy these customizations or extensions in the test environment for verification. When you are ready, repeat the same process to build and deploy your customizations and extensions in the production environment. For instructions, see the *Selling and Fulfillment Foundation: Customization Basics Guide* which includes information about the following topics:

   - How to build and deploy standard resources, database, and other extensions (such as templates, user exits, java interfaces).
   - How to build and deploy Enterprise-Level extensions.

2

Extending Transactions

2.1 About Extending Transactions

You can extend Selling and Fulfillment Foundation programmatically both to enhance the functionality of your implementation of Selling and Fulfillment Foundation and to integrate with external systems.

This chapter describes how to achieve extensibility programmatically, using these mechanisms. Transaction processing mechanisms in Selling and Fulfillment Foundation can be classified into two basic categories:

- Synchronous (on demand) or asynchronous (message driven) services
- Time-triggered transactions

2.2 About Time Triggered Transactions

A time-triggered transaction, or agent, is a program that performs a variety of individual functions, automatically and at specific time intervals. It is not triggered by conditions, events, or user input.

There are three types of time-triggered transactions:

- Business process transactions - responsible for processing day-to-day transactions
- Monitors - watch and send alerts for processing delays and exceptions
- Purges - clear out data that may be discarded after having been processed

You can extend the transactions provided by Selling and Fulfillment Foundation by using one of the following mechanisms:

- User exits
- Event handlers

For information on using the time-triggered transactions provided by Selling and Fulfillment Foundation, see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide*.

2.3 Using User Exits to Extend Standard Transactions

Selling and Fulfillment Foundation provides the ability to extend or override key business algorithms. This is accomplished through user exits that are invoked when the algorithms are run.

A typical user exit can:

- Override application logic by providing its own logic.
- Extend application logic by providing more inputs to the application algorithm.

For example, if an order is split into multiple shipments, you may need to compute the order price differently for each shipment. In order to change the way Selling and Fulfillment Foundation computes order prices, you can override the specific computation or algorithm in the `repriceOrder` user exit.

Each user exit is a separate Java interface. You may choose to implement only those user exits where you want to override or augment the business logic.

> **Note:** When the API or time-triggered transaction is executing the default algorithm, the user exit is called. If your implementation of the function throws an exception, the current transaction is rolled back.

For detailed descriptions of each user exit, see the following packages in *Selling and Fulfillment Foundation: Javadocs*:

- com.yantra.ycm.japi.ue
- com.yantra.ycp.japi.ue

- com.yantra.ydm.japi.ue
- com.yantra.yfs.japi.ue

2.3.1 Implementing and Deploying User Exits

All user exit classes should be deployed as a JAR file that is available in the CLASSPATH of the agent adapter script and in the `smcfs.ear` file deployed in your application server. For more information about implementing user exits, see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide*.

For information on creating and deploying custom classes, see the *Selling and Fulfillment Foundation: Installation Guide*.

2.3.2 Guidelines for User Exits

The following guidelines have to be kept in mind when you are using User Exits within the Selling and Fulfillment Foundation API:

- User Exits are structured to return specific information and hence their usage must be restricted for such purpose alone.
- From the user exits, you cannot invoke APIs that modify the data. This is to ensure that errors do not occur because of the data that is getting modified in the parent transaction (the transaction that calls the user exit), and the same data getting modified in the user exit custom code. For example, you cannot invoke a `changeOrder()` API from a user exit that obtains information related to the same order.

  However, APIs that do not modify the data (like select APIs) can be invoked in the user exits. For example, you can call `getOrderDetails()` API from a user exit.

2.4 Using Event Handlers to Extend Standard Transactions

Selling and Fulfillment Foundation raises events at specific moments in processing. Selling and Fulfillment Foundation enables you to define an action to be performed when a specific event occurs.

In the Selling and Fulfillment Foundation configuration, an event is defined to invoke the Selling and Fulfillment Foundation event handler.

The event handler performs special processing on data published by the event before being transported to the transport services layer.

As part of the event configuration, services can be invoked. When such services are invoked, some of the events, like INVENTORY_CHANGE pass data as a map whereas the other events pass data as an XML. In the event of data being passed as a map, when the service is configured for such an event the data map is converted to an XML as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<YantraXML>
    <XML AccountNo="" AdjustmentType="ADJUSTMENT"
        EnterpriseCode="DEFAULT" ItemID="ITEM1"
        ItemKey="2005030116023851364" ..... />
</YantraXML>
```

Note: By default, when you raise an event that has Map data, the data map gets converted to an XML with 'YantraXML' as the root tag. If you want to override this root tag value, you must set the yfs.sci.event.flow.roottag property value as the new root tag in the <INSTALL_DIR>/properties/customer_overrides.properties file.

For additional information about overriding properties using the customer_overrides.properties file, see the *Selling and Fulfillment Foundation: Properties Guide*.

2.4.1 About Configuring Events

In Release 5.0 and later, you can associate an action to a service to perform special processing required. For example, the event PUBLISH_SHIP_ADVICE could invoke the Selling and Fulfillment Foundation event handler to call a custom Java class. The Java class can augment the publish ship advice XML. The Service Definition Framework can transport the enriched XML data to the transport services layer.

For more information on configuring actions and associating the actions to events, see the *Selling and Fulfillment Foundation: Application Platform Configuration GuidePlatform User Guide for the Console Interface*.

For more information on the various components you use to build a service, see the *Selling and Fulfillment Foundation: Application Platform Configuration GuidePlatform User Guide for the Console Interface*.

The event handlers that can be invoked from an action (configured in the Other tab) are described below.

Note: The following event handlers are provided exclusively for backward compatibility purposes.

2.5 E-Mail Message Event Handler

Selling and Fulfillment Foundation provides a standard Send E-mail event handler for sending e-mail messages on various events. For example, when a line is shipped, you can send shipment information. SMTP-based e-mail is supported for these messages. When configured in the Other tab when configuring an Action, the e-mail is sent synchronously. If the mail server is not available, an exception is thrown. For sending an event notification in the form of an e-mail message using a service, see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide*.

2.5.1 E-Mail Templates

E-mail templates enable you format e-mail messages. The E-mail node receives XML data and merges it with the XSL template you specify. You can configure an E-mail node in the Service Builder to use an e-mail template.

2.5.2 Create Custome E-Mail Template

To create and use a custom e-mail template:

1. Copy the `<INSTALL_DIR>/repository/xapi/template/merged/email/orders_mail.xsl` template file.
2. Modify the file as needed, rename it, and save it within the `<INSTALL_DIR>/extensions/global/template/xsl/<CUSTOM-TEMPLATE-XSL>` directory. You can save it to another directory, but using the

standard directory structure supplied by Selling and Fulfillment Foundation helps ensure consistency.

3. From the Service Builder, configure an E-mail node to use your custom e-mail template. In the Body Template field specify `/template/xsl/<CUSTOM-TEMPLATE-XSL>`.

Note that using the `<INSTALL_DIR>/repository/xapi/template/merged/email/<file_name>.mlt` file with Actions to accomplish the same purpose has been deprecated as of Release 5.0. Use the E-mail node and an XSL file instead.

2.6 Exception Alert Event Handler

Selling and Fulfillment Foundation supports standard event handlers sending exception alerts to the Alert Console. These exceptions are redirected to a specific user or to an exception queue for a group of users. For example, you can send an exception alert to the customer service representative queue when authorization fails so the customer service representative can contact the buyer in person. For more information about the Alert Console, see the *Selling and Fulfillment Foundation: Application Platform User Guide*.

2.6.1 Exception Alert Templates

Exception alert templates enable you to supply additional text to alerts raised. This enables you to make error message more descriptive and easy to understand. They also provide a means of supplying a hyperlink to the resolution screens from the Alert console. For example, for any alert created for an order, shipment, or load document type, a hyperlink is created and displays in the "Created For" column on the Alert List screens. In the Exception Alert Template you can customize this hyperlink or create any other hyperlinks. The input data to the alert node is merged with the template you specify and then posted as the description of the alert raised.

Events that publish data in data buffer format use an ECT template, which are text files that contain tags. These tags are replaced by actual data at run time. All tags use the |#YFS_<tagname>| syntax.

For example, use the tag |#YFS_OrderNo| to have the actual order number appear in its place (if OrderNo is published as a part of the data

buffer). Any of the data elements published in the data buffer can be used in the template.

For the exception console, there are two templates. One template is used for determining the DETAIL_DESCRIPTION field of the YFS_INBOX table. The other template is used for determining the LIST_DESCRIPTION field of the YFS_INBOX table. The templates are merged with the data published (by an event) and the resulting string is populated to the corresponding field.

2.6.2 Create Custom Exception Alert Templates

To create and use a custom exception template:

1. Copy the <INSTALL_DIR>/repository/xapi/template/merged/exception_console/example_exception_console.xsl template file.
2. Modify the file as needed, rename it, and save it within the <INSTALL_DIR>/extensions/global/template/xsl/<CUSTOM-TEMPLATE-XSL> directory. You can save it to another directory, but using the standard directory structure supplied by Selling and Fulfillment Foundation helps ensure consistency.
3. From the Service Builder, configure an Alert node to use your custom exception console template. In the XSL Template field, specify template/xsl/<CUSTOM-TEMPLATE-XSL>.

Note that using the <INSTALL_DIR>/repository/xapi/template/merged/exception_console/<file_name>.ect file with Actions to accomplish the same purpose has been deprecated as of Release 5.0. Use the Alert node and an XSL file instead.

2.7 Publish Event Handler

Selling and Fulfillment Foundation enables publishing of data by the event manager to external systems for interoperability. You can configure any event to publish data to other systems. For example, upon the event ON_SUCCESS of the SHIP_CONFIRM transaction, you may want to publish data to an external financial system.

When configuring the Publish event handler, you must provide a set of system IDs separated by semicolons (for example,

SYSTEM1;System2;TestSystem). These are IDs of time-triggered Transactions that are expected to read and process the data. The event manager writes the data provided by the transaction to the YFS_EXPORT table and writes one record for each system ID configured.

For publishing data using a service, use a Database transport node. For more information, see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide*.

2.8 Execute Event Handler

The Execute event handler tells the event manager to invoke the specified program. Selling and Fulfillment Foundation looks for this program in the PATH. Data is passed to the program as the first command line argument. Like the Publish event handler, the Execute event handler is invoked asynchronously. This is achieved by using a custom API that runs the program in a service.

2.9 Java Extensions as Event Handlers

Java extensions enable you to implement event handlers as Java classes. With each action, you can associate one Java event handler. The Java class you create must implement the `com.yantra.yfs.japi.util.YFSEventHandlerEx` interface. This is achieved by using a custom API which actually runs the program in a service.

2.10 Oracle Stored Procedures as Event Handlers

If you configure an event handler as an Oracle stored procedure, the stored procedure is invoked with the following parameters:

```
PROCEDURE DO_ACTION(TRANID IN VARCHAR2,ACTIONCODE IN VARCHAR2,KEY_DATA IN
VARCHAR2,DATA_TYPE IN NUMBER,DATA_BUFFER1 IN VARCHAR2,DATA_BUFFER2 IN
VARCHAR2,DATA_BUFFER3 IN VARCHAR2,DATA_BUFFER4 IN VARCHAR2,DATA_BUFFER5 IN
VARCHAR2,DATA_COMPLETE IN NUMBER,SHIP_NODE IN VARCHAR2,RETURN_VALUE IN OUT
NUMBER)
```

PL/SQL limits the maximum size of a VARCHAR2 variable to 2,000 bytes, so 2,000 bytes is the maximum length of the string passed in data buffers. If the input is greater than 10,000 characters, the buffer is

truncated and the parameter DATA_COMPLETE has a value of 0 (zero). If the buffer is less than 10,000 bytes and is completely passed to the stored procedure, DATA_COMPLETE is passed as 1.

> **Important:** You must not do a commit or a rollback in the stored procedure at any time in a stored procedure associated to a event handler.

Execution of stored procedures is not a supported service component.

2.11 HTTP Extension as Event Handlers

If you configure an event handler as an HTTP extension, Selling and Fulfillment Foundation sends data using the `post()` function to the specified URL. Data is posted using the variables in Table 2–1.

Table 2–1 Variables Associated with HTTP Extensions

Variable	Description
sTranID	ID of the transaction raising the event.
iDataType	If iDataType is set to 1, XML data is published with the event and sData contains the entire XML string. Otherwise, iData Type is set to 0 and sData contains a name=value pair. To see what the name=value pair contains, see the DBD file associated with the event in the *Selling and Fulfillment Foundation: Javadocs*.
sData	See iDataType description.
sShipNode	Ship node ID, if available.

A service can be configured to post data using the HTTP protocol. For details, see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide*.

2.12 COM Extensions as Event Handlers

Using COM requires setting up your server and runtime clients.

If you define an event handler as a COM extension, Selling and Fulfillment Foundation calls the COM component you specify as the COM object. The COM component you create must expose the IDispatch

interface and a pre-identified function `doAction()` as part of the interface. The signature of the `doAction()` function should be as follows:

```
[IDL Syntax]
doAction ( [in] BSTR sTranID, [in] BSTR sActionCode, [in] BSTR sKeyData,
[in]long DataType, [in] BSTR sData, [in] BSTR ShipNode, [Out] long *Retval);
```

Because Selling and Fulfillment Foundation accesses this function through the IDispatch interface, you can write your COM component either in the Visual Basic or C++ programming language. The function signature that you must expose through Visual Basic is:

```
Public Function doAction(ByVal bsTranID As String, ByVal boActionCode As String,
ByVal bsKeyData As String, ByVal DataType As Long, ByVal bsData As String, ByVal
ShipNode As String, retVal as Long) As Long
```

The function signature that you must expose through C++ is:

```
STDMETHODIMP COMActionImpl::doAction(BSTR sTranID, BSTR sActionCode, BSTR
sKeyData, long DataType, BSTR sData, BSTR Node, long *RetVal)
```

`COMActionImpl` is the class name that implements the `doAction()` function. While configuring the event handler, you must specify the Program ID of the COM component to be invoked.

2.13 Event Chaining and Event Handlers

Some event handlers support invocation of APIs. These APIs can retrieve more data pertinent to the event being raised, or they may direct to perform a business computation or transaction. In turn, the transaction can raise other events, thus resulting in event chaining. For example, one of the event handlers for the ORDER_CREATE event can check the validity of an order and call an API to HOLD the order. This results in the triggering of the HOLD event, which sends an e-mail message to a customer service representative. Event chaining provides an extensibility mechanism that is required for complex business processing.

Important: Actions that are configured to be invoked by the ON_FAILURE event must not perform any database updates. Any database updates performed by an action invoked by the ON_FAILURE event is rolled back.

To access the data published by Selling and Fulfillment Foundation, you must implement the YFSEventHandlerEx() interface, which provides the handleEvent() function to implement as follows:

```
public boolean handleEvent (YFSEnvironment oEnv, String sTranID, String
  sActionCode, Map sKeyData, int iDataType, Object sData, String sShipNode,
  String [] parms) throws Exception
```

Important: While it is possible to implement an API that causes an event that is associated with an action in Selling and Fulfillment Foundation to call the same API, take care to avoid this situation. It creates an infinite loop.

The class name must be configured as the Java object on the Action Configuration screen in the Selling and Fulfillment Foundation transaction configuration. For more information, see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide*.

All the parameters for the handleEvent() function are input parameters, with values inserted by Selling and Fulfillment Foundation. The last parameter in this function is the parms parameter, which is an array of String constants. The values of these string constants are the values of the string constants specified after the class name during the action configuration in Selling and Fulfillment Foundation.

Note: For details on the *YFSEventHandlerEx* interface, see the *Selling and Fulfillment Foundation: Javadocs*.

3

Using Variables for Configuration

3.1 Using Variables for Dynamic Configuration

You can dynamically configure certain properties belonging to actions, agents and services configuration to cut back on your implementation time. For example, you can provide a dynamic way of configuring network properties such as provider URLs, E-mail server, and Sender addresses. These properties can be configured as variables in one place and can be resolved at runtime when these properties are being used.

You can provide variables in place of file or directory names wherever a file name or path can be entered in the Applications Manager. This variable substitution is based on an entry in the `<INSTALL_DIR>/properties/customer_overrides.properties` file and is resolved during runtime.

You can use variable names in the following components of the Applications Manager:

- Service Definition Framework
  - All transport types such as JMS queue name, Initial Context Factory, QCF lookup and Provider URL. File Sender and Receiver, FTP source and destination for sender and receiver directories and HTTP and Webservice transport types.
  - In the e-mail component, where you can specify, the email server, subject, listener port, and From addresses. The dynamic configuration can also be used for specifying the email protocol.
  - Actions
    - In call HTTP extension and Execute Program.
  - Agent criteria details

    - JMS Queue Name, Initial Context Factory, QCF Lookup and Provider URL.

- Printer Devices, Print Documents, and Print Components
- Purge Criteria's log file name
- In the System Management console:
  - You can use variables to specify installation rules such as e-mail server name, server IP address, server listener port, and e-mail protocol.
  - You can provide variables for the JMS monitoring configuration fields, which include: WebLogic Provider URL, WebSphere Channel name, host name, port number and queue manager name.

For the fields identified above, you can configure the values as ${VARIABLE_NAME} in <INSTALL_DIR>/properties/customer_overrides.properties file. It is stored in the database as is and at runtime when the variables are used, a lookup is performed in the customer_overrides.properties file to decipher the value. Since, the values for these variables are fetched from the customer_overrides.properties file, they are specific to a particular JVM.

For additional information about overriding properties using the customer_overrides.properties file, see the *Selling and Fulfillment Foundation: Properties Guide*.

> **Note:** The value of this variable cannot be seen in the health monitor agent details, since the value depends on the JVM on which it is deployed. You have to click on the server details of the monitor agent to view the value of the variable.

For example, if you want to set the File IO Receiver's directory structure to a common variable say ffbase, then the incoming directory should be set to ${ffbase}/incoming. The variable ffbase must be defined in the file as:

```
yfs.ffbase=C:/FileIODir/Receiver
```

This `${ffbase}/incoming` value is stored in the database, and when processing the file adapter, the variable is resolved to `C:/FileIODir/Receiver/incoming`.

The following conditions are assumed for the usage of this variable:

- All the variables when referenced must be in the following format:
  - `${variable_name}`
- All variables should be properly formed. If a variable is not found, no substitution takes place.
- Variables must not contain the '}' character.
- Variables must not begin or end with a whitespace character.
- Templates do not support variables for filenames since they are always resolved within the classpath.

4

Creating Custom Time-Triggered Transactions

4.1 About Custom Time-Triggered Transactions

Selling and Fulfillment Foundation provides infrastructure that enables you to write your own time-triggered transactions. You invoke and schedule these time-triggered transactions in much the same manner as you invoke and schedule the Selling and Fulfillment Foundations standard time-triggered transactions. For information on how to configure the Selling and Fulfillment Foundation standard time-triggered transactions, see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide*.

Depending upon the way time-triggered transactions determine the list of tasks to be processed (their work load), they can be classified into one of the following categories:

- Non task-based (generic) - these time-triggered transactions use custom logic to determine the work they have to perform. They may or may not use the centralized Task Queue.
- Task-based (specific) - these time-triggered transactions use the Task Queue to determine their work.

Selling and Fulfillment Foundation provides infrastructure to create both types of custom time-triggered transactions.

The ability to write non-task-based time-triggered transactions is provided using the `com.yantra.ycp.japi.util.YCPBaseAgent` class. This class provides a generic infrastructure irrespective of whether the Task Queue is used or not and therefore can be used to write any time-triggered transaction.

Task-based time-triggered transactions can be programmed by subclassing the com.yantra.ycp.japi.util.YCPBaseTaskAgent class.

If your transaction is Task Queue based, it is suggested that you use the infrastructure provided specifically to write task-based transactions. This infrastructure automatically determines work for your custom agent from the Task Queue, thus reducing the amount of design and development required for your transaction.

All the custom agents written to Selling and Fulfillment Foundation specification are subclassed from the com.yantra.ycp.japi.util.YCPBaseAgent class, which has two abstract functions. When you implement these functions, they provide the processing capabilities of a time-triggered transaction.

4.1.1 getJobs() Abstract Function

The getJobs()abstract function uses Env, a pre-created instance of a YFSEnvironment object that can be passed to APIs and inXML is an org.w3c.dom.Document object, which contains the input XML. The implementation of this function should obtain jobs (from the database) that need to be run, construct a list of org.w3c.dom.Document objects, and return the list. See the following signature:

```
public List getJobs(YFSEnvironment Env, Document inXML)
```

4.1.2 executeJobs() Abstract Function

Each document in the List returned by the getJobs() function is passed to this function for execution. If this function throws an exception, the exception is logged and the transaction is rolled back, otherwise it is committed. See the following signature:

```
public Document executeJob(YFSEnvironment Env, Document inXML)
```

After all of the documents have been processed by the executeJob() function, Selling and Fulfillment Foundation invokes the getJobs() function again to obtain the next set of tasks that need to be processed by the executeJob() function. This repeats until no more jobs are returned by the getJobs() function.

For examples of the input XML, see the YCPBaseTaskAgent class in the *Selling and Fulfillment Foundation: Javadocs*.

The com.yantra.ycp.japi.util.YCPBaseAgent also provides utility functions for trace logging and timer information with the following signatures:

- public void log(String classMethodName, String message);
- public startTimer(String timerName);
- public endTimer(String timerName);

4.2 Creating Custom Non-Task-Based, Time-Triggered Transactions

Custom non task-based time-triggered transactions should be written as subclasses of the com.yantra.ycp.japi.util.YCPBaseAgent class. For a sample custom time-triggered transaction, see the com.yantra.ycp.japi.util.YCPBaseAgent class in the *Selling and Fulfillment Foundation: Javadocs*.

To write a custom non-task based time-triggered transaction:

1. Subclass com.yantra.ycp.japi.util.YCPBaseAgent.
2. Implement the executeJob() and getJobs() functions in this class.
3. From the Applications Manager, configure a time-triggered transaction and assign an Agent Server to it. For details about configuring time-triggered transactions see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide*.
4. Schedule and run your custom time-triggered transaction according to the instructions in the *Selling and Fulfillment Foundation: Installation Guide*.

4.3 Creating Custom Task-Based, Time-Triggered Transactions

Task-based custom time-triggered transactions are written as subclasses of the com.yantra.ycp.japi.util.YCPBaseTaskAgent class, which is a subclass of com.yantra.ycp.japi.util.YCPBaseAgent with the getJobs() and executeJob() functions already implemented. Creating a task-based custom time-triggered transaction by subclassing this class

involves implementing the executeTask() function to process one task queue record passed to you as input.

> **Note:** If the executeTask() function throws an exception, the exception is logged and the transaction is rolled back, otherwise it is committed.

The logging and timing utility functions available are similar to the ones provided by the com.yantra.ycp.japi.util.YCPBaseAgent class. The signature of the executeTask() function is public Document executeTask(YFSEnvironment oEnv, Document inXML);Env is a pre-created instance of a YFSEnvironment object that can be passed to APIs and InXML is the org.w3c.dom.Document object, which contains the custom task XML. The custom task XML also contains a TransactionFilters Node, which contains all the parameters passed to the task-based custom time-triggered transaction. This node is below the root node of the input XML. For example, see an example for a task-based custom time-triggered transaction.

```
<?xml version="1.0" encoding="UTF-8"?>
<TaskQueue TaskQKey="" TransactionKey="" DataKey="" DataType="" AvailableDate=""
    Lockid="" Createts="" Createprogid="" Createuserid=""
Modifyts="" Modifyprogid="" Modifyuserid="" >
             <TransactionFilters   AgentName=""  TransactionKey=""
           CurrentThread="" NumRecordsToBuffer="" TotalThreads=""/>
</TaskQueue>
```

For a sample custom task-based time-triggered transaction, see the com.yantra.ycp.japi.util.YCPBaseTaskAgent class in the *Selling and Fulfillment Foundation: Javadocs*.

To write a task-based custom time-triggered transaction:

1. Subclass com.yantra.ycp.japi.util.YCPBaseTaskAgent.
2. Implement the executeTask() function in this class.
3. From the Applications Manager, configure a time-triggered transaction and assign an Agent Server to it. For details about configuring time-triggered transactions see the *Selling and Fulfillment Foundation: Application Platform Configuration Guide*.

Schedule and run your custom time-triggered transaction according to the instructions in the *Selling and Fulfillment Foundation: Installation Guide*.

5

External Transactions

5.1 Coordinating with External Transactions

Selling and Fulfillment Foundation provides the capability to run external code either by raising an event, calling a user exit, or invoking a custom API or service. During these invocations, the external systems can begin a transaction. Since the external transaction is not part of the Selling and Fulfillment Foundation transaction it could lead to data inconsistencies if the Selling and Fulfillment Foundation transaction is rolled back but not the external transaction.

External transaction coordination lets the external systems register their transactions with Selling and Fulfillment Foundation. When Selling and Fulfillment Foundation is ready to commit its transaction, the `YFSTxnCoordinatorUE` user exit is invoked, which lets Selling and Fulfillment Foundation handle the commits and rollbacks on the external transactions.

To implement external transaction coordination:

1. Create the custom API that you want to implement your external transaction process, but DO NOT commit the external transactions. Instead, register the external transaction object with the YFSEnvironment by calling the `(YFSEnvironment)env.setTxnObject(String ID, Object txnObj)`.

   The String ID is a unique name used by the user exit implementation class to identify the custom transaction object.

   The following example illustrates a simple custom API.

Example 5–1 External Transaction Coordination of a Custom API

```
public class doSomethingAPI
```

```
{
private YFCLogCategory cat = YFCLogCategory.instance("DoSomethingAPI");

public doSomethingAPI()  // constructor is empty
{
}
public void writeToDB (String key, YFSEnvironment oEnv)
{

try
{
Driver aDriver =
(Driver)Class.forName("oracle.jdbc.OracleDriver").newInstance();
        String url = "jdbc:oracle:thin:@127.0.0.1:1521:qotree2";
        Connection conn = DriverManager.getConnection(url, "Scott",
"Tiger");
conn.setAutoCommit(false);
        String sql = "insert into TxnTest (key) values ('" + key + "')";
        Statement stmt = conn.createStatement();
        stmt.executeUpdate(sql);
        oEnv.setTxnObject("YDBconn", conn);
}
catch (Exception e)
{
System.out.println ("Caught Exception :\n" + e);
}
    }
public Document doSomething(YFSEnvironment env, Document doc) throws
Exception
{
      System.out.println("Executing doSomething method..........");
writeToDB ("doSomething", env);
return doc;
    }
}
```

2. Implement the `com.yantra.yfs.japi.ue.YFSTTxnCoordinatorUE` user exit interface to commit the external transactions either before or after Selling and Fulfillment Foundation perform its commits. Then implement the rollback method so that if the Selling and Fulfillment Foundation transaction triggers a rollback, the `afterYantraTxnRollback(YFSEnvironment oEnv)` method is called to rollback the external transactions as well. Implement the following methods to accomplish this:

- beforeYantraTxnCommit(YFSEnvironment oEnv)
- afterYantraTxnCommit(YFSEnvironment oEnv)
- afterYantraTxnRollback(YFSEnvironment oEnv)

Note: You can use either the beforeYantraTxnCommit or the afterYantraTxnCommit user exit to synchronize commits, depending on your integration requirements.

Calling(YFSEnvironment)env.getTxnObject(ID)enables these methods to obtain the handle to the external transaction object that was previously registered by the (YFSEnvironment)env.setTxnObject(String ID, Object txnObj). Note the ID is the same in both the getTxnObject call and the setTxnObject call.

The following is an example of the YFSTTxnCoordinatorUE user exit interface implementation.

Example 5–2 Sample Interface Implementation

```
public class afterTxnCommit implements YFSTxnCoordinatorUE
{

public void beforeYantraTxnCommit (YFSEnvironment oEnv) throws
YFSUserExitException
{
// before method is not implemented because after method is implemented.
}
public void afterYantraTxnCommit (YFSEnvironment oEnv) throws
YFSUserExitException
{
System.out.println ("Entering method afterYantraTxnCommit.........");
try
{
    Connection ydbConn = (Connection)oEnv.getTxnObject("YDBconn");
    ydbConn.commit();
}
catch (Exception e)
{
System.out.println ("Caught Exception :\n" + e);
}
}
```

```
public void afterYantraTxnRollback (YFSEnvironment oEnv) throws
YFSUserExitException
{
System.out.println ("Entering method afterYantraTxnRollback.........");

try
{
    Connection ydbConn = (Connection)oEnv.getTxnObject("YDBconn");
    ydbConn.rollback();
}
catch (Exception e)
{
System.out.println ("Caught Exception :\n" + e);
}

}
```

3. Launch the Applications Manager and navigate to System Administration > User Exit Management to configure the YFSTxnCoordinatorUE Implementation class as described in the *Selling and Fulfillment Foundation: Application Platform Configuration Guide*.

For more information about the YFSTxnCoordinatorUE user exit interface definition and other detailed information, see the *Selling and Fulfillment Foundation: Javadocs*.

6

Transactional Data Security

6.1 How Is Transactional Data Secured?

Security issues involve controlling who has access to what data, how much they can see, and what they can do with it. Selling and Fulfillment Foundation provides mechanisms that address the following security issues:

- User access control
- Single sign on
- Data encryption

6.1.1 User Access Control

Through an access control mechanism of user group permissions, logins, and order modification permissions, Selling and Fulfillment Foundation provides security measures for multiple levels of customer service and administrative organizations.

APIs control access to different areas of system functionality. It is the responsibility of the caller to ensure that the invoking user has access rights for the function being invoked. Selling and Fulfillment Foundation provides security manager APIs to help in this effort. See the `com.yantra.api.ycp.security` package in the *Selling and Fulfillment Foundation: Javadocs*.

6.1.2 Single Sign On

If a single sign on mechanism is required within Selling and Fulfillment Foundation, implement the interface

com.yantra.ycp.japi.util.YCPSSOManager, which has a getUserData() function that returns a String(UserId). For more detailed information, see the *Selling and Fulfillment Foundation: Javadocs*.

6.1.3 Data Encryption

Encryption ensures that sensitive data is not viewed by unauthorized people. Selling and Fulfillment Foundation provides APIs that enable you to encrypt data such as user names, passwords, and credit card numbers.

In addition, encryption and decryption is only applied after it has been specified within the Applications Manager. For example, only user exits that have been passed credit card information can access decrypted credit card numbers.

6.2 Encryption Logic

Selling and Fulfillment Foundation exposes the com.yantra.ycp.japi.util.YCPEncrypter interface to handle encryption logic. All of the Selling and Fulfillment Foundation encryption and decryption is handled by an encrypter class that implements this interface. This class is specified by configuring the following properties in the <INSTALL_DIR>/properties/customer_overrides.properties file:

- yfs.encrypter.class
- yfs.propertyencrypter.class properties

For additional information about overriding properties using the customer_overrides.properties file, see the *Selling and Fulfillment Foundation: Properties Guide*.

Both classes must implement the com.yantra.ycp.japi.util.YCPEncrypter interface.

The com.yantra.ycp.japi.util.YCPEncrypter interface has the following two functions:

- public java.lang.String encrypt(java.lang.String sData) - sData is the data passed by Selling and Fulfillment Foundation to the implementing class for encryption. The return value is the encrypted string.

- public java.lang.String decrypt(java.lang.String sData) - sData is the data which is required to be decrypted.

For information on writing your own property encrypter class, see the YCPEncrypter interface in the *Selling and Fulfillment Foundation: Javadocs*.

Encryption and decryption functions in this interface are invoked multiple times by Selling and Fulfillment Foundation. Selling and Fulfillment Foundation does not distinguish between clear text and encrypted information. Therefore, the encrypt function may be invoked with previously encrypted data. In order to avoid double encryption, it is important for the encrypt function to be able to distinguish between clear text and previously encrypted information. If previously encrypted information is passed to the function, your implementation of this function should return what is passed into it without encrypting it again.

The decrypt function also should be able to distinguish between clear text and previously encrypted text.

6.2.1 Disabling Encryption and Decryption

To disable encryption (or decryption), implement the encrypt (or decrypt) function to return the same value it is passed as input without any processing.

6.3 Choosing an Encryption and Decryption Strategy

There are multiple deployment options when choosing an encryption strategy. The most typical options are:

- No encryption or decryption
- External tokenization
- Both encryption and decryption
- No decryption

Use the following explanation to guide your decision-making process:

6.3.1 Using No Encryption and No Decryption

If you operate in a secure and trusted environment which is protected physically and electronically and you do not display credit card numbers on the Application Console, you may choose not to implement any encryption logic. Credit Card numbers are be encrypted in this case and are stored in clear text. This is not a recommended option except in the following scenarios:

- Your business does not accept, process, or store credit card numbers or other sensitive information.
- Selling and Fulfillment Foundation passes the externally encrypted credit card numbers. All encryption and decryption is handled externally.

6.3.2 External Tokenization

Sterling Commerce recommends that if you are handling any sensitive payment information, that you tokenize the sensitivie information before passing it into Selling and Fulfillment Foundation.

6.3.3 Using Both Encryption and Decryption

Selling and Fulfillment Foundation encrypts and decrypts credit card numbers automatically as required. Access to clear text credit card numbers is available on the Application Console based on user authorization levels.

6.3.4 Using Encryption But No Decryption

If your business requires Selling and Fulfillment Foundation to store credit card numbers, but you never want Selling and Fulfillment Foundation to automatically decrypt them under any circumstances, you may want to enable only the encrypt function and disable the decrypt function.

This way, Selling and Fulfillment Foundation encrypts the credit card numbers passed in as clear text but never converts them back. Once Selling and Fulfillment Foundation encrypts the information, all your custom extensions are passed as encrypted credit card numbers and

must handle decryption externally. It is important to note that a few user exits in Selling and Fulfillment Foundation (for example, YFSbeforeCreateOrderUE) are invoked *before* the credit card number is encrypted, so it still has access to the clear text number.

6.4 How is Encryption Supported?

Selling and Fulfillment Foundation supports encryption for the following places:

- Properties specified in the yfs.properties, yifclient.properties, and management.properties files
- Credit Card Numbers

6.4.1 Encryption Through yfs.properties

Properties such as the JDBC URL, database User ID and Password can be stored encrypted in the customer_overrides.properties file. Because Selling and Fulfillment Foundation needs this information to connect to the database, these values must be decrypted by Selling and Fulfillment Foundation. If you do not wish Selling and Fulfillment Foundation to ever decrypt data, these properties cannot be stored encrypted.

Note: If you want set any of the properties specified in the yfs.properties file, add an entry for that particular property in the <INSTALL_DIR>/properties/customer_overrides.properties file. For additional information about overriding properties using the customer_overrides.properties file, see the *Selling and Fulfillment Foundation: Properties Guide*.

6.4.2 Encryption for Credit Card Numbers

Selling and Fulfillment Foundation can encrypt Credit Card numbers before storing them in the database. Unlike the properties specified in the yfs.properties file, decrypted credit card numbers are never required by Selling and Fulfillment Foundation for default processing. However, you may extend Selling and Fulfillment Foundation by implementing a user exit that requires decrypted credit card numbers for charging or storing user preferences. If you don't want Selling and

Fulfillment Foundation to decrypt information automatically, you must decrypt these credit card numbers in your implementation of the user exit.

6.5 Encrypting Credit Card Numbers

If you implement encryption, Selling and Fulfillment Foundation encrypts credit card number in the following situations:

- When returned by an API
- When published as a part of event data
- When stored anywhere in the database
- When displayed on the user interface (although the user interface may have an option to override this behavior based on user access)

6.5.1 Encrypting Credit Card Numbers Through APIs

Note: Sterling Commerce recommends that payment information entering the system is already tokenized instead of being encrypted.

Selling and Fulfillment Foundation provides the following APIs that enable you to encrypt and decrypt credit card numbers assuming that both encryption and decryption algorithms have been implemented by the Encrypter class:

- getEncryptedString() - accepts a string passed to it and returns the string encrypted
- getDecryptedString() - accepts an encrypted string and returns the string decrypted
- getEncryptedCreditCardNumber() - returns an encrypted credit card number

Note: getEncryptedCreditCardNumber() has been deprecated in Selling and Fulfillment Foundation, Release 8.5. It has been replaced with getEncryptedString().

- getDecryptedCreditCardNumber() - returns the credit card number that had been encrypted using the getEncryptedCreditCardNumber() API

> **Note:** getDecryptedCreditCardNumber() has been deprecated in Selling and Fulfillment Foundation, Release 8.5. It has been replaced with getDecryptedString().

6.5.2 Encrypting Credit Card Numbers Through User Exits

Only user exits that are passed credit card information can access decrypted credit card numbers. Selling and Fulfillment Foundation provides the following user exits for passing credit card data:

- YFSCollectionCreditCardUE
- YFSCollectionCustomerAccountUE
- YFSCollectionOthersUE
- YFSCollectionStoredValueCardUE
- YFSValidateInvokedCollectionUE

> **Note:** Some user exits, such as YFSBeforeCreateOrder and YFSBeforeChangeOrder, may have access to credit card numbers before they are encrypted in the system.

For detailed information about these user exits, see the *Selling and Fulfillment Foundation: Javadocs*.

6.6 Encryption Through Property Files

Some properties relay sensitive data such as user IDs and passwords, which you may want to encrypt. Any property (except for the `yfs.propertyencrypter.class` property in the `yfs.properties` file) mentioned in the following files:

- `<INSTALL_DIR>/properties/yfs.properties`
- `<INSTALL_DIR>/resources/yifclient.properties` files

can be encrypted using the `<INSTALL_DIR>/properties/customer_overrides.properties` file. For more information about overriding

properties using the customer_overrides.properties file, see the *Selling and Fulfillment Foundation: Properties Guide*.

To encrypt properties:

1. Prefix the property value you want to encrypt with "encrypted:". For example,

   ```
   yfs.dblogin.datasource.name=encrypted:<encrypted value>
   ```

2. Ensure that the security.propertyencrypter.class property is accessible through the CLASSPATH environment variable.
3. Implement the YCPEncrypter interface. For details about this interface, see the *Selling and Fulfillment Foundation: Javadocs*.

These properties starting with "encrypted:" are automatically decrypted at run-time.

Index