IBM

WebSphere Development Studio Client for iSeries: The Integrated Debugger

Inge Weiss   Session id 404596

iweiss@ca.ibm.com   Agenda key 42ML

WebSphere. software

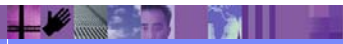*e* business software

IBM

Table of contents

Overview
Service Entry Points
Debugger Functions
Debugging interactive Applications
Launch Configurations and Settings
Runtime Problem Determination
Demo

IBM Software Group | WebSphere software

WebSphere Development Studio

Current 5722-WDS customers with software subscription for V5R3, to upgrade to WDSc V6.0 use feature #: 2656 Available after GA

Upgrade from WDSC 6.0 to 6.0.1 using Rational Product updater

Unlimited Licenses

| RPG | COBOL |
| C/C++ | PDM SEU SDA RLU |

New WDSc Lite
Technology preview

iSeries	iSeries	iSeries	iSeries	Web Facing WDHT support	iSeries Projects	+CODE +VisualAge RPG
Java™	Debug	Struts Web	Web Service		RSE	
JSF	EGL Java generation	Trace	Profiling	DB	XML	App Server	HATS Toolkit

www.ibm.com/software/awdtools/iseries

WebSphere Development Studio Client V6.0.1 based on RWD V6

iSeries Integrated Debugger   © 2006 IBM Corporation

There is now only one application development product sold by IBM, for iSeries, as of V4R5. This is WebSphere Development Studio (Development Studio), which includes all four host compilers, all traditional tools (ADTS = PDM+SEU+SDA+RLU+DFU+AFP+CGU), and unlimited licenses of the workstation-based toolset named WebSphere Development Studio Client (formerly WebSphere Development Tools).

If you are an existing customer who has a subscription, you can upgrade to Development Studio free of charge. Without a Software Subscription, there is an upgrade fee. New licenses of Development Studio are priced very competitive compared to the combined prices of all constituent products. As of V5R1, there is no way to purchase the compilers or tools individually. So if you have RPG at V5R1 or higher, you must have Development Studio and hence are entitled to Development Studio Client.

For consultants who do not have an iSeries of their own, but still wish to have the client tools, Development Studio Client is also made available as a passport advantage product so it can be purchased "off the shelf" from IBM Direct.

Development Studio has been a huge success, with over 80,000 licenses sold. Just as every development machine used to have PDM and SEU, every development machine will now have all the modern Application Development tools from IBM. This ubiquity is especially important for business partners who build and sell software. These Business Partners are now free to build software using any of the technologies or tools in Development Studio, and can assume their customers will have the tools required to tailor everything from RPG to Java and Web user interfaces. This effectively raises the lowest common denominator to a level unparalleled by any other operating system.

The debugger in WebSphere Development Studio contains a common debug component with an iSeries specific part added to it. This debugger is also part of WDSc Lite.

There are actually multiple debuggers included. The type of application and where you run it, determines which debug engine is started.

Since all debuggers use the common user interface, they all look very much alike. There are differences in the functionality though, depending on the system the program is running on. This presentation will concentrate on the Integrated iSeries debugger, the one that is invoked from the Remote Systems view.

WebSphere Development Studio Client Advanced Edition contains the same debugger as the standard edition.

Integrated iSeries Debugger - Overview

- RPG, Cobol, CL, C, and C++
- ILE and non ILE, incl. free-form RPG
- DB2 and SQL stored procedures
- Source and Listing view
- Batch, interactive, and Multi-Threaded Applications
- Client/Server Applications
- Distributed Applications

Debugging an application always requires that the programs are compiled with the appropriate debug options.

For ILE, you can dynamically switch between source and listing view, provided that you compiled your program with Debug view *ALL.
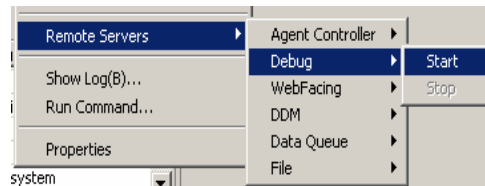
IBM

Debugger Functions - Overview

- **Manage program execution**
  - Step/Run commands
  - Multiple breakpoint types
- **Work with content of variables**
  - Display
  - Modify
- **Work with content of storage**
  - Display
  - Modify for teraspace enabled programs

In a typical debug session, you want to control the execution of the program by setting breakpoints and using the different Step and Run commands. You also need to be able to look at the content of the variables and possibly change them.

Start/Stop Debug Server on iSeries Host

- From RSE, right click a subsystem (iSeries Objects, iSeries Commands, iSeries Jobs, or IFS Files).
- Select Remote Servers > Debug > Start to start and Stop to stop.

Remote Servers ▶	Agent Controller ▶	
	Debug ▶	Start
Show Log(B)…	WebFacing ▶	Stop
Run Command…	DDM ▶	
	Data Queue ▶	
Properties	File ▶	
system		

- Or, from 5250 session, issue command STRDBGSVR to start and ENDDBGSVR to stop.
- Once per iSeries system
- Default port 3825
- WRKSRVTBLE add or change the entry QDBGSVR2

To be able to start a debug session from the workstation, the debug server must be started on the iSeries host. This can be done from the RSE or a 5250 session. The debug server remains running until it is stopped from the RSE or 5250.

The debug server listens on default port 3825.

Use command WRKSRVTBLE to add or change the entry QDBGSVR2

7

Debugger Invocation

Debug As
Program
iSeries job
Debug (Prompt)
Program
Service program
Debug (Service Entry)
ILE Program
Service program
Module
Procedure
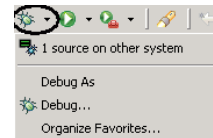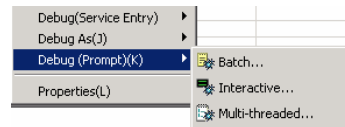
To start debug session, select entry in Remote Systems view and use popup-menu, workbench Run menu, debug tool button, or debug pull-down menu.

You can debug programs from the Remote Systems view, the iSeries Table view, or the workbench toolbar in three ways:

•In a batch job

•In an interactive job

•In a server job (multi-threaded)

IBM

Where does the Application run?

Debug As…

Batch	Submitted to batch
Interactive	5250 emulation, STRRSESVR
Multi-threaded	Creates BCI job
Job	Specified job

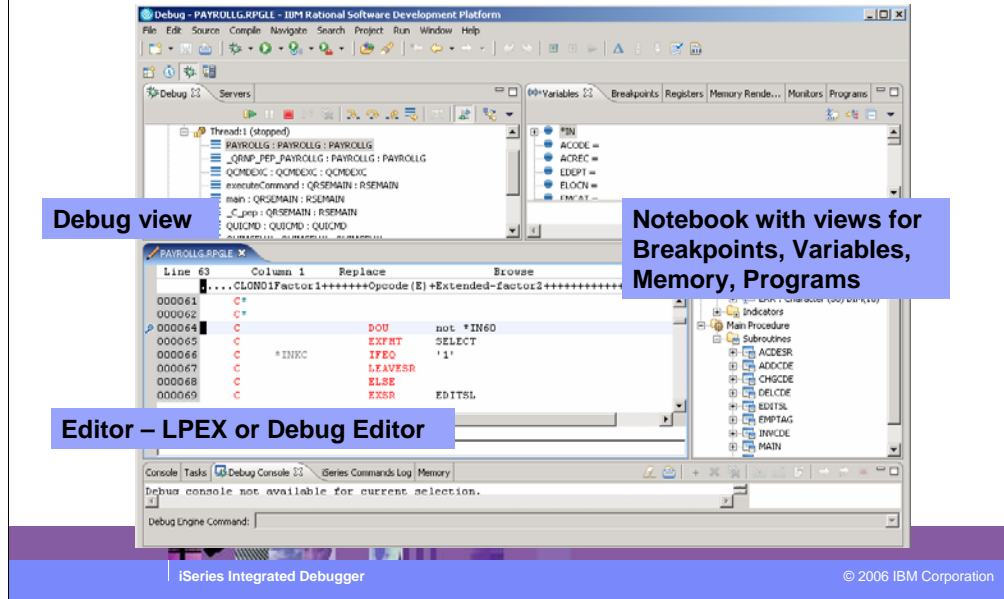Depending on the debugging mode you selected, the application will run in a specific job.

Debugging in batch creates a new job, submits it to the batch job queue with Hold option, releases the job, stops at the first executable line and passes control to the user.

Debugging Interactive requires a 5250 emulation session where the STRRSESVR command has been run.

For multi-threaded applications, the debugger creates a Batch Immediate (BCI) job and calls your application in that job.

Selecting Debug as job allows the user to attach to a job on the iSeries. This can be a batch, interactive, multi-threaded or even Java job.
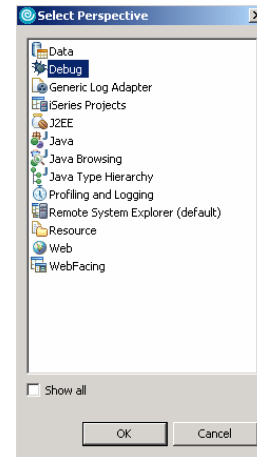
Debug Perspective

Debug view

Notebook with views for Breakpoints, Variables, Memory, Programs

Editor – LPEX or Debug Editor

© 2006 IBM Corporation

The Debug perspective contains the tools and views to debug a program. It opens when you start the debugger. Here you see the call stack and source view. The top right contains a notebook with several other views, for example, Breakpoints, Monitors, Programs.

Debug perspective

- **Collection of editor and views**
- **Users can customize the perspective**
- **Views can be re-sized and re-positioned through drag and drop**
- **Views can be closed**
- **Views can be added**
- **Use Save Perspective As… to save customized perspective**

Like any other perspective in the workbench, the Debug perspective can be customized. Once you have a layout that matches your work style, you can save the perspective under a new name and use it in all debug sessions from then on.
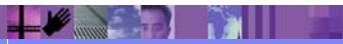
Table of contents

Service Entry Points

- Service Entry Points for ILE on V5R2 or later
  - You know the program you want to debug but the program gets started in a server environment
  - You know the userid the program will be running under
  - You don't know the job name where the program will be running

Service entry points are a special type of breakpoint that can be set on any ILE program V5R2 or later, which is not currently under debug.

You use service entry points when you wish to debug an application that makes use of the Toolbox or multiple jobs. Examples of cases where you would want to use a service entry point include:

•Applications that are invoked by a Toolbox program call. In this case, you would set a service entry point in the program that will be called by the Java application. When the program is called and the code where the service entry point is set is about to execute, the debugger can take control of the application and stop at that line. With this technique, you can put the program which is invoked by the Toolbox, under debug when you do not know which job it will be running in.

•Programs that are spawned by other programs. In this case, you would set a service entry point in the application that will be spawned. When the program is spawned and the line where the service entry point is set is about to execute, operation will be suspended and the debugger will be able to gain control of the program and stop at that line. When a service entry point is set, it is triggered when the application not currently under debug is called.

Service Entry Points (continued)

- Great for:
  - WebFaced applications
  - Web applications
  - Toolbox calls
  - Any program you want to debug

You can debug all sorts of applications such as any batch program, a WebFaced application or a Web application.

You cannot call your applicatoin in the RSE job, otherwise the RSE job would not be available for any other actions for the duration of the debug session.

Setting Service Entry Points

To start a debug session:

- **From the Remote System Explorer**
  - In the Remote Systems view, select the program, service program, module, or procedure
  - In the pop-up menu, select Debug (Service Entry) > Set Service Entry Point
  - Run the application from anywhere, except RSE server job
- **From the Service Entry Points view**
  - In the pop-up menu, select Set, or press the toolbar button
  - Fill in the Set Service Entry Point dialog
  - Run the application from anywhere

From the Remote Systems view, you can set Service Entry Points for programs, service programs, modules or procedures.

If you set a Service Entry Point on a program, service program, or module, it will be set on all procedures in the selected object. A Service Entry Point set on a procedure, is valid for that specific procedure only.

All Service Entry Points that are set from the RSE are listed in the Service Entry Points view.

> Note: Whenever the program gets invoked from the specified userid (by default the one used for the connection), the program will be stopped at the Service Entry point and the Integrated Debugger on your workstation will start a debug session for the specified program.

To set a service entry point from a Debug session, start a debug session for the program. Select **Add Service Entry Point** from the pop-up menu of the source or the prefix area for the line where you want your program to stop. This will invoke the **Add Service Entry Point** dialog box, which displays the program, module, source file, and line number of the service entry point that will be created. In this dialog box, specify the user profile for which the service entry point will be activated. By default, the user profile is set to *CURRENT (the user profile for the current debug session).

15

Service Entry Points view

Service Entry Points view is automatically displayed when Service Entry
Point set.

Otherwise display from
Windows > Show view > Other > iSeries > iSeries Service Entry Points
- List of all Service Entry Points set from the RSE
- Pop-up menu to work with Service Entry Points
    Set, Modify, Remove, Enable/Disable
- Filter window
- Pull down menu to Clear all or selected Service Entry Points

Library	Program	Program Type	Module	Procedure	User ID	Connection	Enabled
RSELAB10	PAYROLLG	*PGM	*ALL	*ALL	IWEIS5	torasbcc	Yes
RSELAB10	CLR1	*PGM	*ALL	*ALL	IWEIS5	torasbcc	Yes

Remote System Details | Tasks | iSeries Table View | iSeries Commands Log | iSeries Service Entry Points

All Service Entry Points that are set from the RSE are listed in the Service Entry Points view.

A pop-up menu allows you to set new Service Entry Points, modify existing ones, remove or enable and disable them. Toolbar buttons for these actions are also available.

From the pull down menu, you can clear all Service Entry Points. This will remove them and also end the server program that gets started when you set a Service Entry Point. The server program will also end, when you exit the IDE.

16

Service Entry Points -Refresh

- Service entry point is not valid after program is recompiled.

- Since 6.0, a Refresh button in iSeries Service Entry Points view to re-set the service entry points.
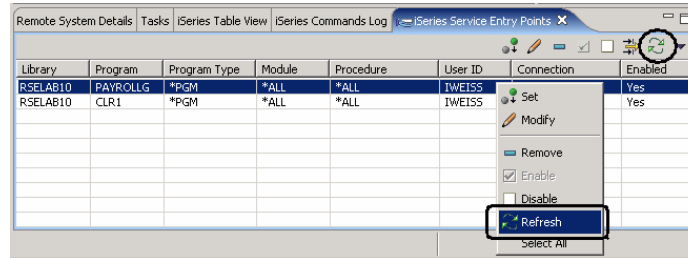
A Service entry point is specific to a program object. When you change the source and re-compile a program on which a service entry point was set, the service entry point is not valid any more. You can use the Refresh action from the pop-up menu of the service entry point to re-set it, by selecting Refresh.

17

IBM

Table of contents

Debug Perspective

Debug - PAYROLLG.RPGLE - IBM Rational Software Development Platform

File Edit Source Compile Navigate Search Project Run Window Help

Resume, Suspend, Terminate, Step Into, Step Over, Step Return

Debug Servers

Call Stack

My iSeries program [iSeries: Debug Interactive Application]
State: <stopped> Connection: iSeries(TM) (TORAS5B8CL:3825)
Thread:1 (stopped)
PAYROLLG : PAYROLLG : PAYROLLG
_QRNP_PEP_PAYROLLG : PAYROLLG : PAYROLLG
CLR1 : CLR1 : CLR1
_CL_PEP : CLR1 : CLR1
QCMDEXC : QCMDEXC : QCMDEXC

Variables Breakpoints Registers Memory Rende

**Variables
Breakpoints
Monitored variables
Programs under debug**

*PGM RSELAB01/CLR1
CLR1
CLR1
CLR1
*PGM RSELAB01/PAYROLLG
PAYROLLG
PAYROLLG
PAYROLLG

CLR1.CLLE PAYROLLG.RPGLE

Line 64   Column 1   Replace   Browse

....CLONO1Factor1+++++++Opcode(E)+Factor2+++++++Result++++++++Len++D+Hi

```
000061    C*
000062    C*
000064    C            DOU      not *IN60
000065    C            EXFMT    SELECT
000066    C    *INKC   IFEQ     '1'
000067    C            LEAVESR
000068    C            ELSE
000069    C            EXSR     EDITSL
000070    C            END
```

Current Line

Break Point

Outline
Global Definitions
Files
Fields
Indicators
Procedure
Subroutines

**Add Properties:
Window ->
Show View ->
Other ->
Basic**

Console Tasks Properties Debug Console iSeries Commands Log Memo

Property	Value
Call Level	16
Call Level Number	6187

Here we see the common Eclipse Debug perspective, which is being used to debug an RPG program. The common debug user interface has been connected to the iSeries debug engine since Version 5.0, to offer a common and compelling debug story for OPM/ILE RPG and COBOL and CL, and ILE C and C++.

In the upper left pane is the call stack, much like option 11 in the i5/OS WRKACTJOB. It shows the calls that reflect your current program execution. When you double click an item in the stack, its source (if available) is shown in the source pane in the middle. The upper right is where all the various views are for working with data contents, breakpoints etc.. The middle is the debugger source view, with source executable (debug) lines in blue, others in green. The current line of execution is highlighted, and breakpoints appear as a dot with a check mark in the left margin.

Although not part of the common Debug Perspective, the Properties view contains valuable information about the selected object, which could be a breakpoint selected in the Breakpoints view, an entry selected in the call stack, etc. To add the Properties view, click on the menu item **Window** and **Show View** on the pull down menu, select **Other** from the submenu, expand **Basic**, select **Properties** and click OK.

19

Program Execution

- **Step Into**
  – Debug the next call level
- **Step Over**
  – Run the next call level and stop at the next statement
- **Step Return** (for ILE on V5R3 or higher)
  – Run until you are back in the previous call level and stop at the next statement
- **Resume**
  – Run until an event is encountered
- **Run To Location**
  – Run and stop at the current cursor position or until an event is encountered
- **Suspend**
  – Halt program at point of execution
- **Terminate**
  – End the debug session

Step Return allows you to step out of a called program, module, or procedure. It runs the remainder of the program, module, or procedure and stops at the statement following the call.

Suspend allows you to get control back when your program is running on the System i host. This could be, for example, a looping program or a program waiting for input from a 5250 screen.

20

IBM

Setting Breakpoints

Ways to set a Line Breakpoint:

➢Double click in prefix area
➢In Editor prefix area, select Add Breakpoint from pop-up menu
➢In Editor, select Add Breakpoint from pop-up menu
➢In Breakpoints view, select Add Breakpoint >Line from pop-up menu.

Ways to set a Watch Breakpoint:

Double click variable and
➢In Editor, select Add Watch Breakpoint from pop-up menu
➢In Breakpoints view, select Add Breakpoint > Watch from pop-up menu.
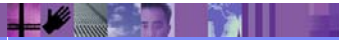
You can only set breakpoints at executable lines.

Conditional Line Breakpoint

Add a Line Breakpoint

Required information

Sets a breakpoint to stop execution at a specific s

☐ Defer breakpoint until executable is loaded

Program

`*PGM RSELAB01/CLR1`

Module

`CLR1`

Views
- ● *SOURCE
- ○ *LISTING
- ○ *STATEMENT

Source(optional):

`CLR1.CLLE`

Line:

`8`

< Back   Next >

Add a Line Breakpoint

Optional parameters

Make the breakpoint conditional upon the following parameters

Thread: Every ▼

Frequency
From: 99
To: Infinity
Every: 1

Expression: &COUNT > 98

< Back   Next >   Finish   Cancel

Set Frequency
Value to start
Value to stop
Breaks to skip

Set Expression
Conditions must be true
for break to happen

Complex expressions for
C, C++ only

You can also set conditional breakpoints. The frequency allows you to limit the number of stops. Specifying an Expression will only stop program execution when the condition is true. The type of expression allowed depends on the programming language.

Watch Breakpoint

Add a Watch Breakpoint

Required information
Sets a breakpoint to stop execution when data changes at a specific address

Address or expression: AR3

Number of bytes to watch: 0

< Back | Next > | Finish | Cancel

Number of bytes
to watch set to 0 for
defined length of variable

Maximum length 128

Maximum of 256 watches
per system
Variable spanning a page
break uses 2 watches
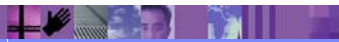
You can also set conditional breakpoints. The frequency allows you to limit the number of stops. Specifying an Expression will only stop program execution when the condition is true. The type of expression allowed depends on the programming language.

Variables view

- In V5R3 or later, local variables support is available for ILE RPG and ILE COBOL programs, for C and C++ also in earlier releases.
- Variables view automatically displays all variables of current scope.
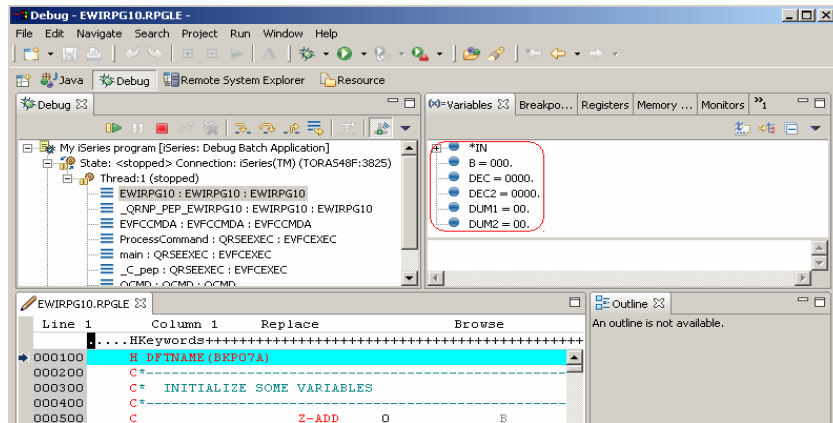
Displaying local variables for CL is not supported.

Filtering Local Variables

- Local variables can be filtered to show only user defined variables or pre-defined variables in ILE RPG or ILE COBOL.

iSeries Integrated Debugger   © 2006 IBM Corporation
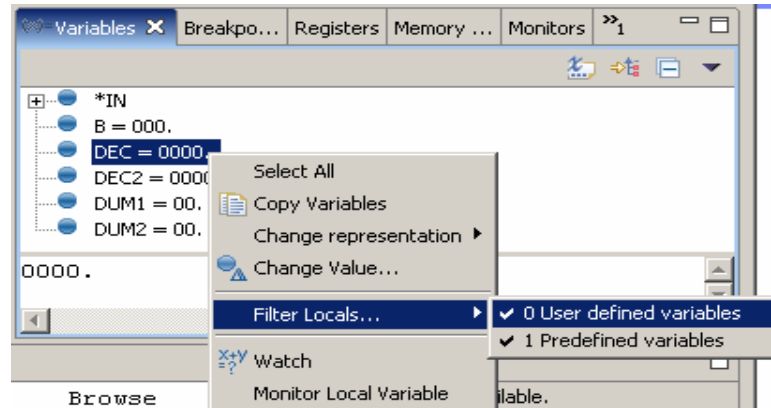
You can use Filter Locals to select not to display either the user defined variables or the predefined variables.

To monitor a variable, select it in the source by double clicking and use the Monitor Expression menu option.

Monitor Expression Dialog

1. Select variable
2. Click plus sign for dialog
3. Variable pre-filled
4. Can add dimension

• Data in Monitor view updated at each stop
• Changed data marked in red with triangle icon
• Double click on entry to open for edit, modify value and press Enter

iSeries Integrated Debugger

© 2006 IBM Corporation

You can also monitor a variable from the Monitor Expression dialog, which is available from the Monitors view. The dialog is pre-filled with any variable that is selected in the source. This dialog is especially useful when you want to monitor one specific array element or an element of a structure.

Display Memory Content

1. Select variable to display in a Memory view
2. Select Monitor Memory > Memory
3. Displays hex content
4. Select Monitor Memory > ASCII/EBCDIC to display text

- Each variable gets a new tab
- Memory displayed starting with address of selected variable
- Edit for teraspace enabled pgms

To display the storage starting with the address of a selected variable use the Monitor Storage menu option.

Note: Teraspace enabled programs will allow you to modify the storage content.

Programs view

Click plus to add a Program, Service Program or Java Class

Pop-up menu to remove programs from debug

Expand program entry and double click to display source

iSeries Integrated Debugger

© 2006 IBM Corporation
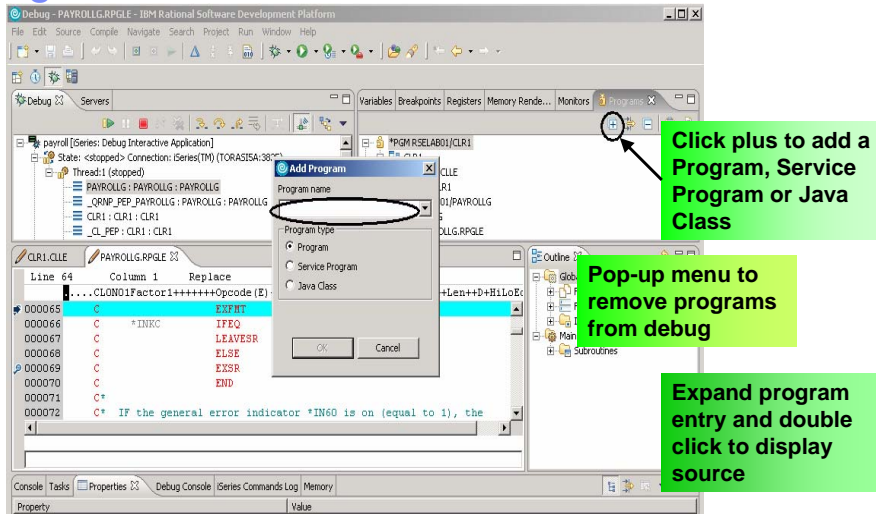
The Programs view lists all programs, modules and procedures of the current debug session. You can use the Add Program dialog to add programs, service programs and Java classes to your debug session. Click the plus sign to bring up the dialog. The pop up menu of a program, service program or Java class allows you to remove the selected entry from the debug session. The initial program and the one you are currently stopped at cannot be removed.

Double clicking on a source or procedure entry displays its source.

iSeries debugger Help
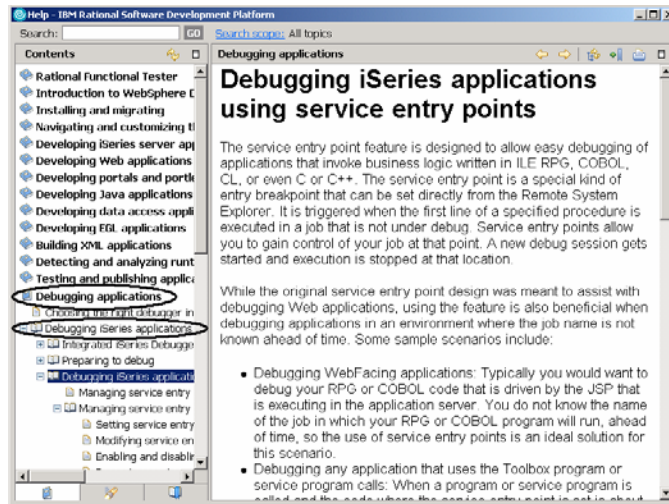
F1 help is available from the different views.

Select **Help > Help Contents** to look at the table of contents. The section **iSeries application development** contains the information about the Integrated Debugger.

IBM

Table of contents

31
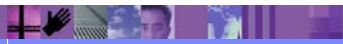
Debugging Interactive Applications

Interactive debugging requires 5250 session and STRRSESVR command
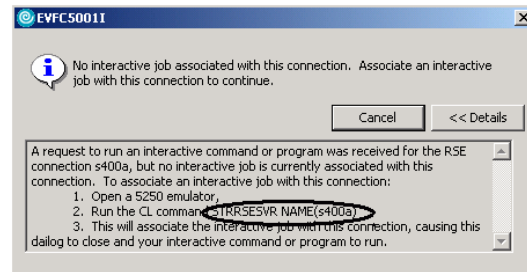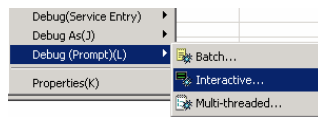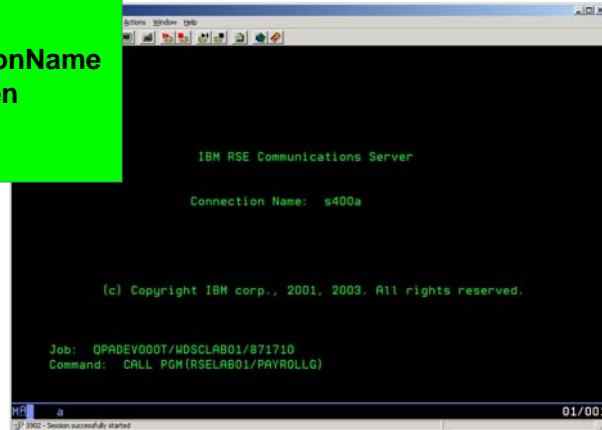
If you selected Debug as Interactive but there is currently no interactive RSE session, a message is displayed informing you that there is no such session and also giving you instructions how to remedy the situation.

Tip: You can copy the command from the message (STRRSESVR NAME(yourServerName)) and paste it into the 5250 command line.

Once the STRRSESVR command has been run, your 5250 session is associated with the RSE server and blocked from other use. The screen will look similar to the one above.

If you want control over the 5250 session back, use the pop up menu of one of the RSE subsystems (iSeries Objects, iSeries Jobs, iSeries Commands or IFS Files) and select 'Release Interactive job'.

Program I/O on Emulation Window
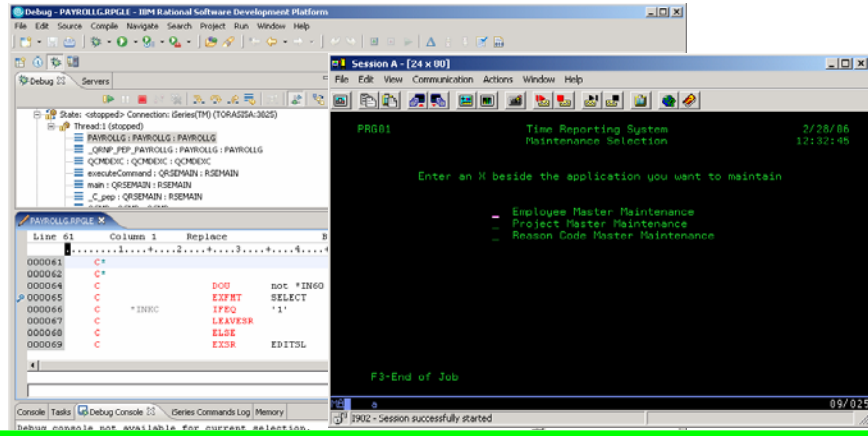
Switch to emulation window for screen I/O
Note: Terminating the debug session does not terminate the interactive program

When you run an interactive program, the program waits for input from the 5250 emulation session.

Library List of RSE Session

- **Initial library list from user profile**
- **Change for current RSE session from RSE pop up menu of Library List**
  - Add Library List Entry
  - Change Current Library
- **Change in Properties from RSE pop up menu of any subsystem. Used next time you connect**
  - Initial Library List node
    - Add Library List Entry
    - Change Current Library
    - Re-order library list entries

Changes to the library list affect the RSE job.

Changes to the library list specified in the Properties of a connection, take effect every time the connection is activated. If you change the properties, these changes become available the next time you codnnect.

Tip: You can create multiple connections to the same iSeries host and set different properties for each one.
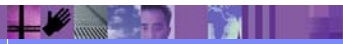
Library List of Debug Session

- **Debug Batch or Multi-threaded**
  Library list of the current RSE session

- **Debug Interactive**
  Library list of interactive job + libraries set in Properties
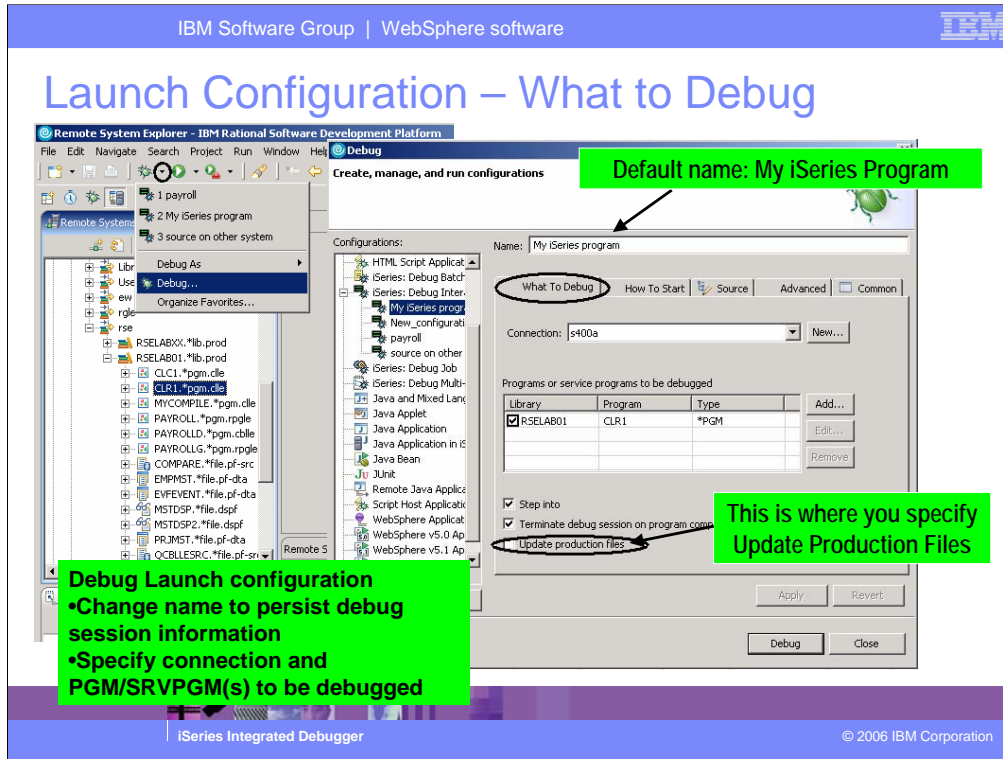
- **Debug job**
  Library list of the job

The debugger uses the library list information from the RSE session for jobs that are created by the debugger.

For interactive jobs, the library list needs to be set before the STRRSESVR command is run. Libraries specified in the properties of the selected connection will be added to the library list of the intereactive job.

Table of contents

Overview
Service Entry Points
Debugger Functions
Debugging interactive Applications
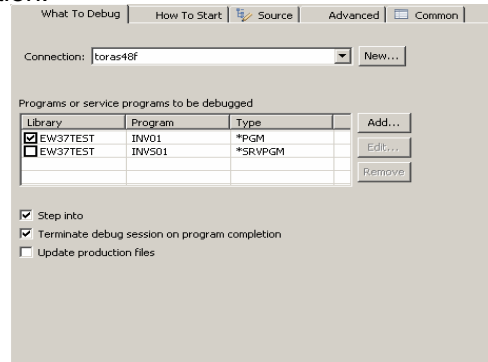**Launch Configurations and Settings**
Runtime Problem Determination
Demo

Launch Configuration – What to Debug

IBM Software Group | WebSphere software

Default name: My iSeries Program

Debug Launch configuration
•Change name to persist debug session information
•Specify connection and PGM/SRVPGM(s) to be debugged

This is where you specify Update Production Files

iSeries Integrated Debugger

© 2006 IBM Corporation

You can start the Debugger in several ways: directly from the Remote Systems view using Debug As, or from the Launch Configuration dialog. Starting directly from the RSE without prompt does not allow you to specify parameters to be passed to the program. Prompting and using the Launch Configuration dialog allows you to modify how the program is invoked including to specify parameters.

When you invoke the debugger from the pop up menu, a default launch configuration called **My iSeries program** is created with the information of the selected object.

Specify multiple PGMs/SRVPGMs

- Can select multiple programs or service programs to be debugged in the Remote Systems view or add to list in launch configuration.

| What To Debug | How To Start | Source | Advanced | Common |

Connection: toras48f

Programs or service programs to be debugged

Library	Program	Type	
☑ EW37TEST	INV01	*PGM	Add...
☐ EW37TEST	INVS01	*SRVPGM	Edit...
			Remove

☑ Step into
☑ Terminate debug session on program completion
☐ Update production files

You can specify multiple programs or service programs for the debugger, but you have to specify the initial program.

You can also select multiple programs/service programs in the RSE and select debug.

Launch Configuration – How to Start
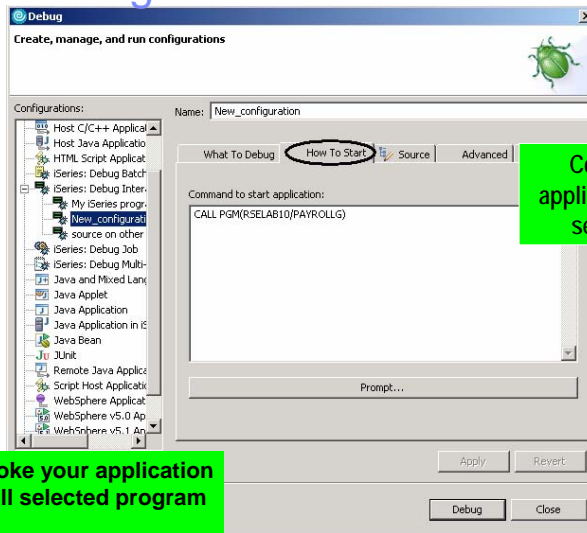
In the How To Start page of the Launch Configuration you specify the command that invokes your application. By default, this page contains a call command for the selected program. You can modify this command to add parameters or you could specify a different command or program that invokes your application. You can use the Prompt button to get a prompt dialog for the specified command.

For a batch program, a SBMJOB command with a call to the specified program is the default.

40
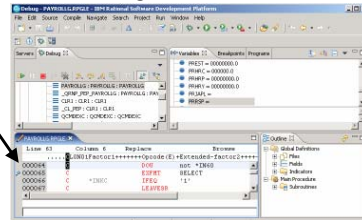
The Source Locator allows users to add new iSeries source physical files and/or IFS folders on any connected iSeries host to search for the source members during debug.

Launch Configuration – source location

➢If source remains in the original library (compilation time), debugger will find it

➢If source member has been moved since compile occurred, use iSeries source file source locator (next slide)
   ✓Moved to a different library
   ✓Resides on a different iSeries host

➢If IFS source is moved, use Remote Folder source locator.
   ➢Moved to a different folder
   ➢Resides on a different iSeries host

➢Source locator can be specified from Source tab of launch configuration

If the debugger is unable to find the source, the user can specify a new source location, or select to have a Listing view displayed, if available.
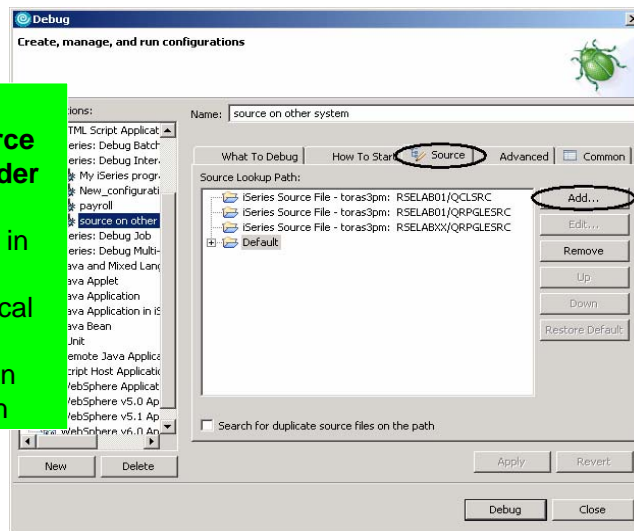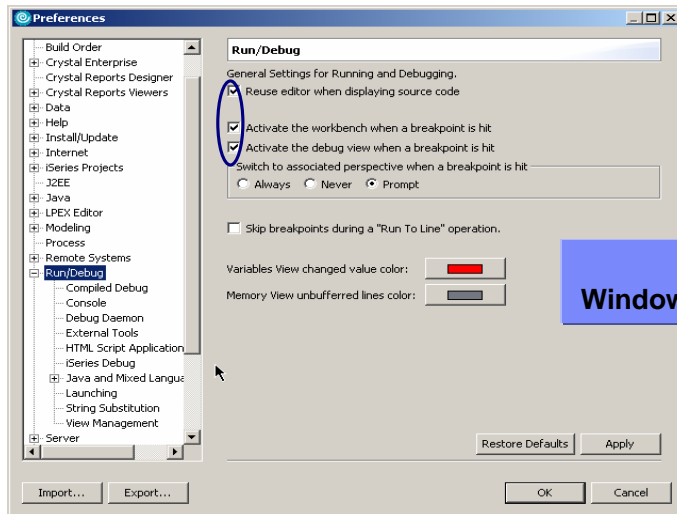
The Source tab contains the source locator.

Debug Preferences

Select:
Windows -> Preferences

When using the debug editor, check the 'Reuse editor when debugging source code' check box if you want only one tab for multiple source in the debug editor. Switching to a different source can then be done from the call stack or the Programs view. If de-selected, each source will have its own tab and selecting its tab will display the source.

Check the 'Remove terminated launches when a new launch is created' check box, to delete the messages in the call stack that belong to terminated debug sessions. That way only the currently active sessions are listed in the call stack.
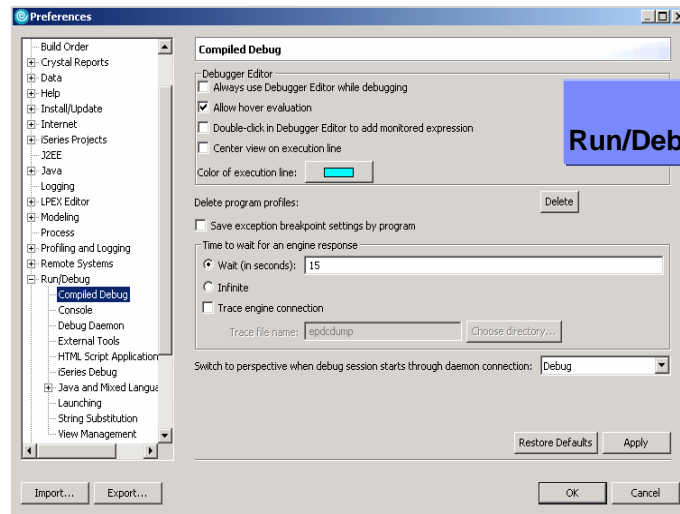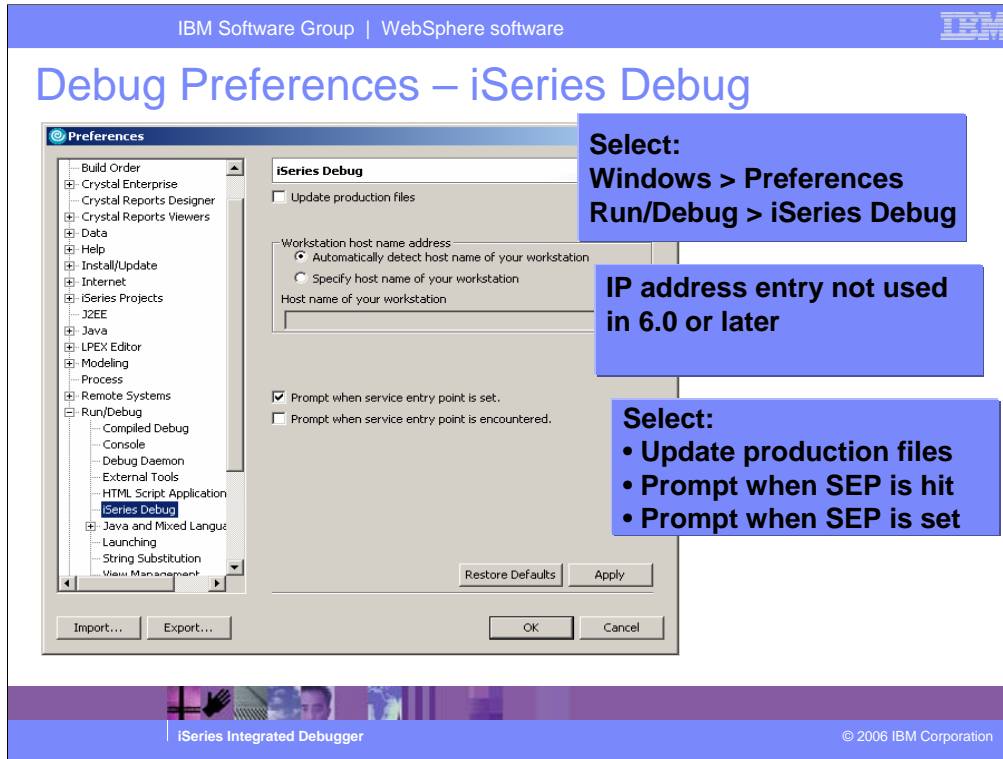
44

Un-checking the 'Always use Debugger Editor while debugging' check-box will use the LPEX editor to display the source while debugging.

Debug Preferences – iSeries Debug

Select:
Windows > Preferences
Run/Debug > iSeries Debug

IP address entry not used in 6.0 or later

Select:
• Update production files
• Prompt when SEP is hit
• Prompt when SEP is set
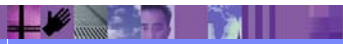
iSeries Integrated Debugger   © 2006 IBM Corporation

By default, Update production files is not selected. Check the check box to allow the debugger to update production files. You need to set this preference if you are using service entry points to start your debug session and want to be allow update production files.
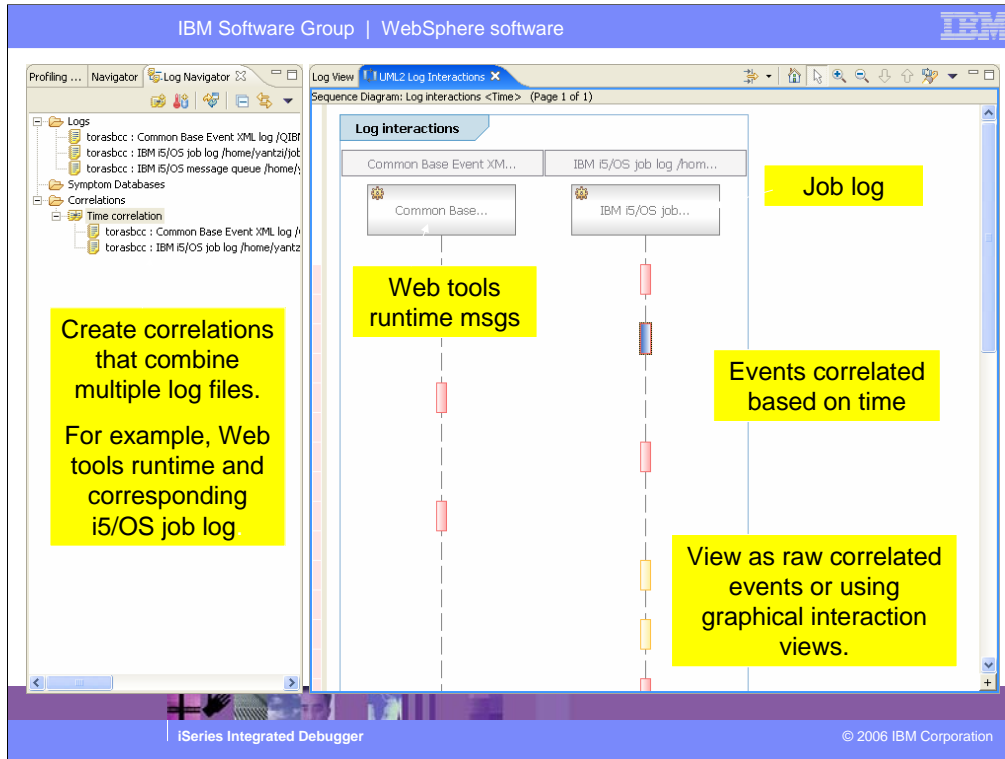
If your workstation has more than one IP address, the debugger may not be able to resolve the workstation's host name or IP address. In this case, select Specify host name of your workstation and enter either the host name or IP address in the entry field.

Selecting Prompt when service entry point is encountered will display a dialog to allow you the choice of starting a debug session or not.

IBM

Table of contents

47

IBM Software Group | WebSphere software

Log interactions

Common Base Event XM...   IBM i5/OS job log /hom...

Job log

Web tools runtime msgs

Create correlations that combine multiple log files.

For example, Web tools runtime and corresponding i5/OS job log

Events correlated based on time

View as raw correlated events or using graphical interaction views.

iSeries Integrated Debugger   © 2006 IBM Corporation

For some runtime problems, you don't even know which program you need to debug.

Correlating runtime messages from WebTools or WebFaced applications with the corresponding job visualizes what happens at what time in your application. If an error occurs, it helps you to isolate, which component is causing it and which program you need to debug.
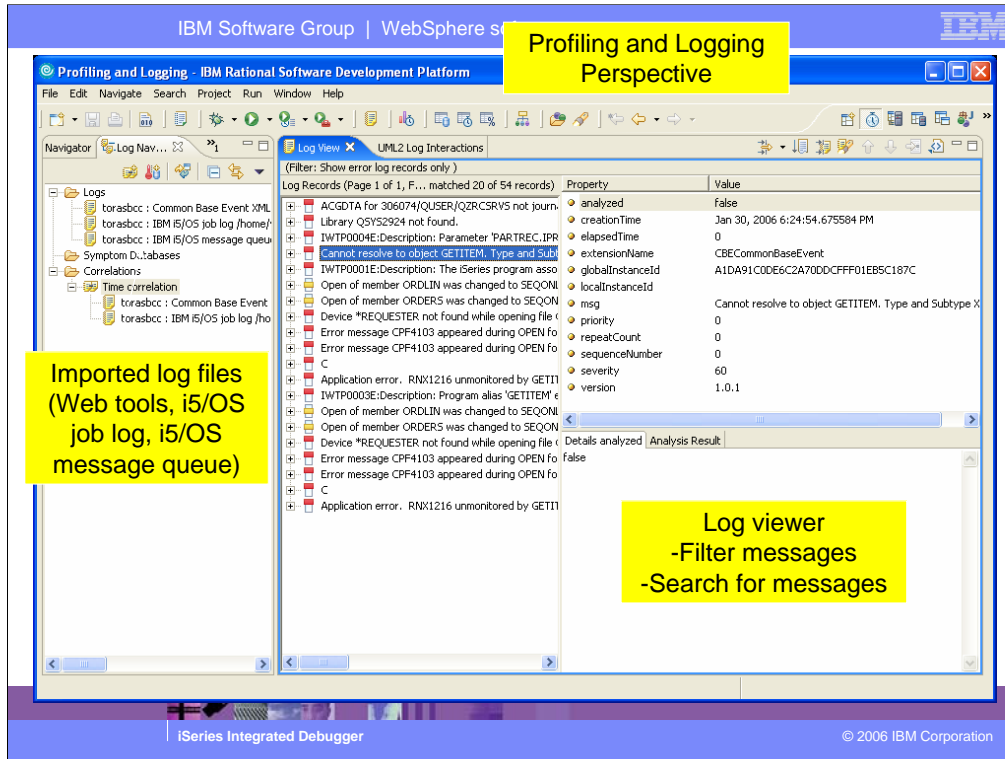
The Profiling and Logging perspective allows you to import i5/OS logs and message queues.

You can search for messages and define filters for example to show only error messages.

IBM

Problem Determination

- Switch to industry standard Common Base Event (CBE) format for WebFacing and Web Tools runtime error messages
  Use the Import > Log File wizard
  Select Common Base Event XML log
  Customize logging in the files:
    webtoolslogging.properties for Web projects
    webfacinglogging.properites for WebFacing projects

- Adapter for converting i5/OS V5R4 messages (job log, message queue) to CBE format
  Advanced edition only
  Use the Import > Log File wizard
  Select "IBM i5/OS message queue" or "IBM i5/OS job log"

- Use Log and Trace Analysis tools to import, view and correlate error messages from composite (multi-tiered) applications

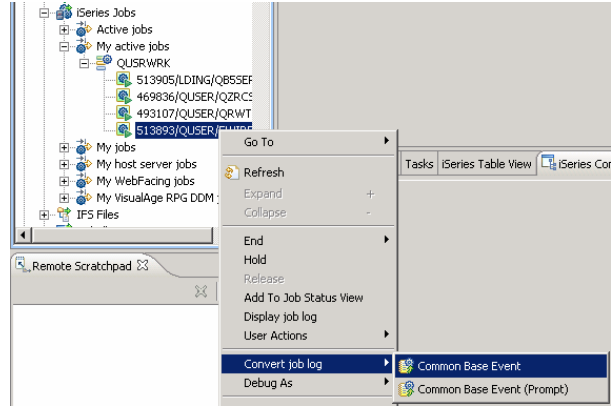This is the first release of WebSphere Development Studio that uses the CBE format.

Convert Job Log

i5/OS V5R4

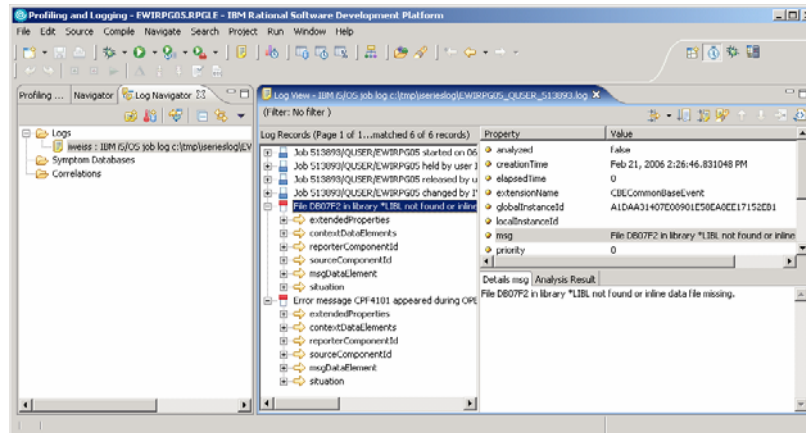WDSc Advanced edition
    only

Invoke from RSE

Download log to
    workstation
    and import into
    Profiling and Logging
    Perspective

You can convert a job log directly from the RSE.

Log and Trace Analyzer – Profiling and Logging

Message queues can only be converted from the Profiling and Logging perspective.
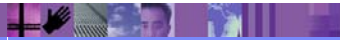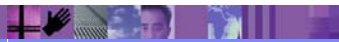
Table of contents

53
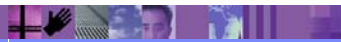
Summary

- Simplify problem determination
- Source level debugger
- Fully integrated
- Common user interface
- Improved developer productivity
- Service Entry Point to debug Web Apps

Additional Information

- **homepage:**
  http://ibm.com/software/awdtools/iseries
  - **Select Library link for Labs, Tutorials, Presentations**
- **newsgroup:**
  news://news.software.ibm.com/ibm.software.
  websphere.code400

Additional Information – WDSCI Mailing List

- **To post a message email: WDSCI-L@midrange.com**

- **To subscribe, unsubscribe, or change list options: http://lists.midrange.com/mailman/listinfo/wdsci-l or email: WDSCI-L-request@midrange.com**

- **Before posting, review the archives: http://archive.midrange.com/wdsci-l**
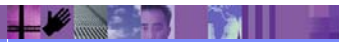
Trademarks and Disclaimers

The following terms are trademarks of International Business Machines Corporation in the United States, other countries, or both:

AS/400	e-business on demand	i5/OS
AS/400e	IBM	OS/400
eServer	IBM (logo)	System i5
@server	iSeries	

Rational is a trademark of International Business Machines Corporation and Rational Software Corporation in the United States, other countries, or both.
Intel, Intel Logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.
Linux is a trademark of Linus Torvalds in the United States, other countries, or both.
Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.
UNIX is a registered trademark of The Open Group in the United States and other countries.
Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.
Other company, product or service names may be trademarks or service marks of others.

Information is provided "AS IS" without warranty of any kind.

All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics may vary by customer.

Information concerning non-IBM products was obtained from a supplier of these products, published announcement material, or other publicly available sources and does not constitute an endorsement of such products by IBM. Sources for non-IBM list prices and performance numbers are taken from publicly available information, including vendor announcements and vendor worldwide homepages. IBM has not tested these products and cannot confirm the accuracy of performance, capability, or any other claims related to non-IBM products. Questions on the capability of non-IBM products should be addressed to the supplier of those products.

All statements regarding IBM future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. Contact your local IBM office or IBM authorized reseller for the full text of the specific Statement of Direction.

Some information addresses anticipated future capabilities. Such information is not intended as a definitive statement of a commitment to specific levels of performance, function or delivery schedules with respect to any future products. Such commitments are only made in IBM product announcements. The information is presented here to communicate IBM's current investment and development activities as a good faith effort to help with our customers' future planning.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.

Photographs shown are of engineering prototypes. Changes may be incorporated in production models.