

Royal Cyber Rational EM Development Modernization Newsletter

Technical content for a Rational world

Articles

- “Editor Wars”, Chris Leland, Page 1
- “RTC EE – Introduction”, Bill Napolitano, Page 5
- “HATS/RCP →← RDz Integration”, Juzer Ali, Page 7
- Royal Cyber in the News, Page 13

Setting Up an Exemplary
Excellence In
IT Industry by winning

- Smarter Commerce Sell Award 2013
- Impact Mobile Innovation Award 2013
- IBM Business Partner Beacon Award 2013

[Learn More](#)





Editor Wars: LPEX vs. the COBOL and PL/I Editors

By Chris Leland, Royal Cyber

If you listen to the old guard at IBM bluster about using LPEX/ISPF vs. the COBOL Editor* you'll hear things such as:

- Because COBOL is a fixed-LRECL block style language, it benefits from block (think the Prefix Area) commands for editing – provided by LPEX in ISPF mode. And because Java/C++/HTML etc. are free-form languages use an Eclipse-style editor**.
- Asking z/OS guys to learn:
 - New IDE + New procedures/work processes + New vocabulary/terms + New SCM interfaces – and new REXX/CLIST interfaces +
 - New Data Editing and SQL development tools + new ISPF 3.x tools (esp. Search) + escalation in mouse-development skills ...
 - And to top it off – asking programmers to give up 10, 20, 30, 40 years of ISPF Edit mastery seems hardly an intelligent on-ramp for veteran developers.
- Show me where the Filters and Autocomment features are in the COBOL Editor
- Until RDz completely replaces ISPF we'll have to maintain our ISPF (including =2/editing) skills. So how is this easier?
- LPEX/ISPF has hot-key support for almost everything, etc.

And if you listen to **new-age developers** – or early adopter types – or perhaps RDz experienced/power users you'll hear this about why the COBOL Editor is superior:

- All of the above is just emotional push-back. The COBOL Editor is clearly superior technology – it's barely Street Legal.
- And it's the future of editing tools/technology.
- And it provides an on-ramp to Java/C++/HTML/SOA coding projects.
- Show me where really cool features like collapsing paragraphs/sections, and auto-formatting your source code are in LPEX/ISPF
- And would you really do block-style editing with LPEX? (short answer – not really – at least in the current LPEX release)
- And isn't Hex editing far easier with the COBOL Editor? (short answer – absolutely, yes)

Naturally, each of these viewpoints is valid.

- The more change you impose on veteran technologists – especially developers for whom software is a means to an end (i.e. making the rent money) – the more likely they are to reject RDz – citing initial loss of productivity. By teaching the LPEX/ISPF editor this issue is reduced.
- For anyone who's used/used to (and likes) Eclipse-style editing, productivity in text/free-form languages OR COBOL, JCL, PL/I, Assembler, etc. the COBOL Editor will be a better choice.
- The LPEX/ISPF editor supports almost all - but not all (not 100%) of ISPF functionality. O note – there's no support for ISPF Edit Macros in either editor.
- The COBOL Editor contains more advanced functionality (see **Table x**).
- However having more advanced functionality is a double-edged sword:
 - It's great to have extended functionality - as new skills are assimilated and new options discovered for productivity/ease-of-work
 - But "more" can be more daunting – even more intimidating to veteran TSO developers - who are often quite happy with their typing-oriented ISPF editing craft (thank you very much)
- Starting in v9.0 RDz added a simple Context Menu "**Open With...**" function that provides one click switching, between LPEX/ISPF and the COBOL Editor and vice-versa. Let's see if this new simple "swap my editor" ends the RDz Editor Wars.

Our view

Stay focused on both the human factors and "big picture" of your Rollout. If your veteran COBOL programmers are taking their first run at RDz do everything you can to max-out their confidence in using new, modern tools – by onboarding them with LPEX/ISPF. After doing a deep-dive editing/usage modeling survey – used to establish your (not IBM's) out-of-the-box defaults for the RDz editor workspace preferences. For Entry-Level programmers or anyone who's seen carrying around a book on Eclipse, definitely start them off with the COBOL Editor.

After a few months of RDz use you should introduce the veteran programmers to the COBOL Editor. Some will jump all over it – others will not. Which is fine, as not every z/OS developer is dying to use a fully tricked-out "Barely Street Legal" editor. Some of us get to work (quite happily) in a minivan.

* Because there is considerably less functionality in the LPEX/ISPF for PL/I than for COBOL (or even for LPEX/Assembler) – the "PL/I Editor" is a much more beguiling prospect. So that war more of a skirmish.

** We @ Royal Cyber are very interested in discussing the premises that block/coding style editors are better for COBOL, JCL, and Assembler. And free-form text editors are better for Java/HTML/C++, etc. And using ISPF for HTML or a Java-style editor for COBOL is an impedance mismatch.

Feature/Function Table – RDz v9.0.1

Feature – Function	LPEX/ISPF	COBOL Editor
ISPF Prefix Area Commands	Y - Support for line/block copy/paste/move, exclude, insert/delete/repeat, etc.	N – No support
ISPF Command Line/Commands	Y – Foundation “native ISPF” commands – but no support for Edit Macros.	N – No support
Ctrl/F (Search)	Y – High-quality search functionality. Better functionality than COBOL editor	Y – Windows-like Search – Overall not as powerful as LPEX but allows for HEX search (not in LPEX)
Block Copy/Paste	Y	Y – Superior to LPEX
Hex Editing	Y	Y – Superior to LPEX
Hover – Mouse-over exposes variable declaration	Y	Y – Hover also exposes Copybooks
Occurrences in Compilation Unit	Y	Y
Program Control Flow – Full graphical view of program execution semantics	Y	Y
Data Element table	Y	Y
Filter View	Y – Provides high-value COBOL analysis functionality	N
Auto-comment code	Y – Can auto-comment statement modification	N
COBOL Numbering Support	Y	Y
Collapse Sections/Paragraphs	N	Y
Synchronize with Outline view	Y	Y – v9.0.1 provides a slick “quick outline” popup – inside the code
Integration with Menu Manager	Y	Y
Integration with SQL/Data Studio	Y	Y
Cut/Copy/Paste	Y	Y
Refactor	Y	Y
Integration with Preprocessor	Y	Y
Identify unreachable code	Y	Y
Comment/Uncomment	Y	Y
Upper-Case (support for ISPF CAPS ON)	Y	N
Content Assist	Y	Y
Bookmarks, Task View and Named Marks	Y	Y
Multiple window'd editing sessions	Y	Y
Split/Screen view	Y – Can split screen (Ctrl/2) and can use Multiple Editing sessions	N – But can simulate with Multiple Editing sessions on same window

Local History	Y	Y
Software Analyzer/Code Review	Y	Y
Identify unreachable code	Y	Y
Color coded variable usage (highlights modified versus referenced variables)	N – But can see modified vs. referenced with Occurrences in compilation unit	Y
Perform Hierarchy – Tree view of your program's execution semantics	Y	Y – And integrated with Hover
Format Source Code – Indents according to shop coding standards	N – Unfortunate that this is not supported. Source Formatting is extremely useful	Y
Quick Fix – Single-click a variable name to select/fix syntax errors	N	Y
Show source of selected element – Isolates paragraph or section code	N	Y
Open COBOL declaration	Y – PF-Key or Context Menu driven	Y – Hover/Context Menu driven
Surround With – Provides the ability enclose selected statements IF, Perform...	N	Y
Open copybooks	Y	Y – And can peek at, and open w/hyperlink
Save Actions – Allows you to define/standard actions upon file save	N	Y
Scope-terminator visualization	N	Y
Show matching brackets – For conditions	Y – Ctrl+M	Y – Mouse-click

Next steps...

RDz is one of the primary strengths of Royal Cyber. In 2014 IBM chose Royal Cyber to manage its RDz “Distance Learning” program - out of all the RDz business partners world-wide. We offer deep and quality service engagements in every phase of RDz – from Installation and Deployment, to Training/Mentoring and adoption, to administering and supporting both the RDz client and server – and finally, to evaluating your RDz Adoption and Return-on-Investment.

- For details on the Royal Cyber RDz Rollout and project/task usage modeling contact us: inquires@royalcyber.com
- To sign up for Royal Cyber RDz Free Distance Learning: <http://royalcyber.com/royal-cyber-rdz-distance-learning-training-schedule/>
- To get a copy of the January 2014 Newsletter – including additional deep-dive articles on RDz and RAAi: <http://royalcyber.org/mn/index.html>

Chris Leland; Chris is Royal Cyber's primary RDz technical instructor - having worked full-time with RDz for almost four years; developing applications in COBOL, teaching RDz classes and mentoring in COBOL and Assembler, installing and configuring RDz, delivering custom workspace design sessions, writing an RDz evaluation exam, and integrating RDz with Rational Asset Analyzer. Chris passed the IBM/RDz Certification exam with flying colors in 2011.



By Bill Napolitano

Think back now... back when you were in software training (university, graduate school, technical college, high school ... whatever). I'll bet you dreamt about a someday-future filled with such boffo undertakings as: Working late on a Friday to deliver project documentation, going through two bags of Doritos while updating application versioning information, staring at green-screen history reports and listings – until your eyes are so red you need Visine to pass a sobriety test. No? I got that wrong? You'd actually rather write and test creative software algorithms to solve business and technical problems than waste your cycles on administrivia?

Can't say that we're surprised. So then it's time — **now** not later - to take a long, hard look at your Source Control Management (**SCM**) processes and products with an eye towards moving off systems that were state-of-the-art during the 1970s/1980s. It's time to modernize your SCM and Project Management tools, as well as your SDLC processes, and to move forward with an integrated, repository-platform based solution... for example, Rational Team Concert EE.

What is Rational Team Concert EE?

Rational Team Concert (sometimes called Team Concert, or RTC for short) – is a Jazz-repository-based product from IBM/Rational that, through shared Eclipse technology and hyperlinking integrates software project design, development, testing and delivery activities such as: Planning/Project Management, Work Items, Source Control, Build and Management Reporting.

This comprehensive “All-in-1” integrated system supports (just about) all of your application lifecycle management requirements – and provides the following benefits relative to the 1980's SCM model:

- Extensive reduction in manual processes and procedures
- Traceability from software versions to project Work Items and developer tasks
- Reduction in the number of disparate tools and products used to manage all phases of software project delivery
- Cross-platform coordinated/automated application build (compile/link/deploy the Java and CICS or IMS components of an application in a controlled manner)
- Ability to store and version modern media assets not supported by z/OS VSAM file I/O (that which underpins many of the popular MVS SCM systems)
- Real-time/Hyper-link integration between your SCM and Build functionality, Construction Tools (such as RDz*), Testing tools (RDz Integrated Debugger), Analytics (Rational Asset Analyzer*) etc.

The “**EE**” – which stands for Enterprise Extensions, includes additional functionality designed to enhance management of z/OS and IBM i based applications. RTC EE includes::

- Dependency detection
- Dependency driven build (Build what is affected by a change)
- Change driven Promotion and Deployment
- Sparse Workspaces (use dependency information so developers only load what is needed for a change)
- Context Aware Searching – (natural language search for Work Items and source)

RTC Enterprise Extensions are available as part of the RTC Developer for IBM Enterprise Platforms license. RTC EE can coexist with existing mainframe SCM tools allowing users to benefit from the planning and work item management whilst preparing a phased move to RTC's own SCM. The icing on the cake is that RTC software and maintenance fees are – depending on which mainframe SCM product you're currently running – typically up to 50% lower.

“This all sounds fantastic!” you say. “So then what's the catch?” (There's gotta be a catch) Well, besides installing RTC EE, you will need to:

1. Get the code out of your current SCM
2. Migrate the build scripts used to compile/link the various configurations of run-time (language), database, transaction system and environment. Upwards of one hundred separate and distinct (and complex) scripts are often needed
3. (possibly most ambitious) - You will have to reengineer your current SDLC processes – changing what has constituted (up to) decades of arcane procedures built into and stemming out of the limitations of your current (1980's) tooling.

** For additional information on Rational Asset Analyzer's integration with RDz and RDz's Integrated Debugger see the Royal Cyber Development Modernization Newsletter Volume 1.



“Oh” - you say – exhaling. So maybe now’s not quite the time to reinvent that 1980s SCM wheel? Except – that there’s almost no thread left on the tires, and as a result you’re getting terrible gas mileage due to all that mind-numbing and often redundant administrivia.

Okay... let’s take another look at those three requirements listed above. And we’ll do this by bringing in an expert on RTC EE; Reader? Meet Chris Trobridge, the IBM Rational Product Manager for RTC EE;

Getting the source code out of your SCM

Royal Cyber: Chris, as RTC EE Product Manager, what are a few **keys to success** for source code-migration-to-RTC challenges?

Chris Trobridge: *It is essential the migration team has application expertise because:*

- *Project and component structures need to be well-defined before migrating code to RTC. Optimally the migration team will include a person who knows the application well and the aid of a tool like RAA to understand the application structure and ensure structures are sensible. This step is essential to get the most out of RTC EE features.*
- *No matter where code is currently held the migration process will uncover issues. Missing pieces, incompatible, duplicate, unknown, unidentifiable code and unbuildable applications are some of the things you may find. You must allow time and have the expertise on hand to resolve these issues. The migration team must include an application expert for each application as it is brought into RTC.*

Migrating the Build Scripts

Royal Cyber: When shops are migrating Build Scripts to RTC what’s the single most important consideration?

Chris Trobridge: *This requires knowledge of RTC EE and the SCM tool being replaced. There is no substitute for experience here and for many organizations it is an opportunity to do housekeeping on their build processes.*

Chris Trobridge, IBM Rational Product Manager for RTC EE

Reengineer SDLC Processes

Royal Cyber: What could a company using traditional green-screen SCM tooling expect to get out of RTC by way of improved SDLC processes?

Chris Trobridge: *RTC opens up tremendous possibilities for SDLC improvements. RTC offers support for LEAN, Agile and traditional processes, allowing mainframe teams to work collaboratively and harmoniously with distributed developers. RTC also enables productive practices like continuous integration/testing to be considered.*

Our take

You could – of course – continue to deliver z/OS application functionality using a retro-SDLC, limited to the SCM processes sanctioned by your 1980’s application lifecycle management tools. And – consequently, you could continue to inhale Doritos as you squander your creative cycles doing superfluous project administrivia.

But, at some point your shop decided to move off of punch cards and on to command line TSO – then to full-screen ISPF – then (if you’re lucky) to RD - just as you moved your business application stack from Assembler to COBOL, etc. Because at some point it becomes clear-cut that using contemporary technologies aren’t just better on paper – they save time & money.

And certainly there are those in any organization who are regressive. And while listening to all points of view is a necessity, at some point it **will** become painfully obvious that the constraints and limitations of an obsolete application lifecycle-plus- tools is indefensible. And when that time arrives, you’d do well to take a long hard look at RTC EE. We at Royal Cyber not only consult in it (in our customer-facing projects) - we also use it internally to manage our solution engagements. If you’d like to discuss any facet of our experience with RTC EE feel free to email us: inquiries@royalcyber.com

A wise man once said, “*When the facts change so does my opinion.*” To many, the verdict about whether a 1980’s SCM scales to the requirements of today’s application development requirements is in. They’ve moved on to RTC EE. When will you?

Bill Napolitano is a freelance z/OS Application software and tools author. Our thanks to Chris Trobridge, Rational IBM for his time and expertise with the above article.

RDz Integration with z/OS Process

By Juzer Ali, Royal Cyber

Effective z/OS Integration Using HATS/RCP + RDz's Menu Manager Functionality

As a modern IDE, RDz is often purchased as a replacement for TSO/ISPF – even though IBM is careful not to oversell expectations. It's good that they don't – because no two TSO/ISPF customer installations/toolsets are alike – anywhere, in the world (trust us on this). So, while RDz provides considerable uplift to the z/OS programmer's maintenance and production support tasks - and is best in class for Enterprise Modernization tooling - there will always (yes ... **always**) be some measure of ISPF green screen development applications/tools that RDz will not support. These include: CLISTs and REXX Execs, custom ISPF Dialog Manager applications developed in-house, and 3rd Party purchased ISPF applications.

And during any RDz Rollout you'll be forced to make some decisions about how to access these green screen dev-apps from RDz. You can:

1. Get to them from Host Connection Emulator in RDz
2. Utilize Menu Manager – RDz's built-in scripting language for accessing TSO commands
3. Write Java/Eclipse plug-ins that access the RDz Remote Systems Explorer API-set
4. Or integrate these applications using HATS/RCP

Our view...

Here are some of the trade-offs and some of the considerations for choosing your integration approach:

- **Host Connection Emulator** – works of course, but did you really purchase RDz to provide 3270-based tooling for your developers? Wasn't the goal to bring the benefits encoded in 40+ years of software/hardware engineering to the workplace? (please say "yes")
- **Menu Manager** is a simple little scripting language – embedded in RDz, but it only provides access to single request/response TSO commands; including CLISTs and REXX Execs – but no ISPF (3270 screen) Dialogs – and no ISPEXEC commands/keywords. I
 - In our experience, most CLISTs and REXX Execs are implemented as multi-step conversations with a user – and many include ISPEXEC operations. Access is verboten from Menu Manager
- **Java/Eclipse plug-ins** are not so limited, but are typically quite expensive to build out, and must be maintained and tested with each new release of RDz (especially releases when IBM jumps to new Eclipse frameworks (RDz 9.0.x – which utilized Eclipse 4.2... up from 8.5.x which utilized Eclipse 3.6).
- **So that leaves HATS/RCP** – which is our recommended integration platform for RDz access to z/OS and to your indispensable:
 - **Complex CLISTs and REXX Execs**
 - **ISPF 3270 tooling**
 - **3rd Party products**
 - **Custom ISPF Dialog Manager applications**
 - **And even access to IBM/ISPF tooling that RDz does not support**
 - See example below of several SDSF actions not available from the RDz JES functionality

What is HATS/RCP?

Host Access Transformation Services or HATS for short, is an IBM/Rational packaged product – that comes free-of-charge, with RDz and that offers a seamless interface from RDz/Eclipse to 3270-based ISPF Dialogs executing in the TSO/CICS/IMS environments.

The RCP (Rich Client Plugin) functionality is available through HATS and it provides a graphical development experience that is aligned with the RDz/eclipse GUI framework, including but not limited to:

- **Context Menus** - often driven from Menu Manager actions) can be used to launch HATS applications
- **HATS application "green screen scraping"** - dynamically accessing TSO/ISPF and integrated into RDz and rendering 3270 output in Eclipse View/Perspective GUI format
- **Simplified integration features** – including single sign on capabilities

HATS/RCP applications make use of 3270 streams in the background. The HATS development package (called the **HATS Toolkit**) allows you to customize the look-and-feel, as well as the behavior of 3270 green screen applications.

Besides access to TSO/ISPF from RDz/Eclipse, a few additional benefits of using HATS/RCP include:

- You can modify what the 3270 screen looks like when you access it
- You can remove unwanted fields of the screen and hide them
- You can also perform/automate some screen navigation so you don't have to type in keystrokes manually:
 - For instance, in order to get to the SDSF panel through green-screen, you have to:
 - Log into ISPF and type the SDSF command on the ISPF panel,
 - And then access the SDSF main panel.
 - Using HATS you can go directly take an RDz user to the SDSF main panel through a single mouse-click without having to log on to the system and typing anything at all.
- Using HATS/RCP from RDz, you can simultaneously be logged on to multiple SYSPLEX(s) and/ or CICS or IMS regions.
- The HATS/RCP toolkit can be used to customize the green screens completely or the default transformation of green screens can be leveraged.
- Because most z/OS SCM products access ISPF panels, HATS/RCP can be used to integrate z/OS SCM functionality with RDz. And seamless integration with your SCM from RDz is one of – if not the most important integration element in your RDz workbench.
 - Full disclosure; RDz's CARMA framework is a better fit for CA-Endevor, but for access to Serena Changeman, CA-Panvalet, CA-Librarian, Alchemist, and home-grown applications you'll want to look at HATS/RCP.

HATS/RCP development and deployment

The process begins with a HATS/ RCP developer, who creates a **HATS Rich Client Project** using the **HATS Toolkit**.

- Again, for developing HATS/RCP integration functionality from RDz to z/OS the HATS Toolkit is a free product from IBM. There are no run-time license fees, and no server interaction.

A HATS Rich Client Project is - in essence, an Eclipse plug-in project - this is the RDz/Eclipse connection. Once developed and tested, the project is **exported** as a HATS rich client application – it then becomes a standard Eclipse plug-in that can be installed into RDz – using either Installation Manager or using the Install New Software... option off the RDz Help menu.

This HATS client application – typically - contains the logic to accept the RDz user's id, password, and other relevant information and uses this information to navigate to the relevant TSO/ISPF Dialog – behaving like a 3270 terminal under the covers. This is done using various HATS macros.

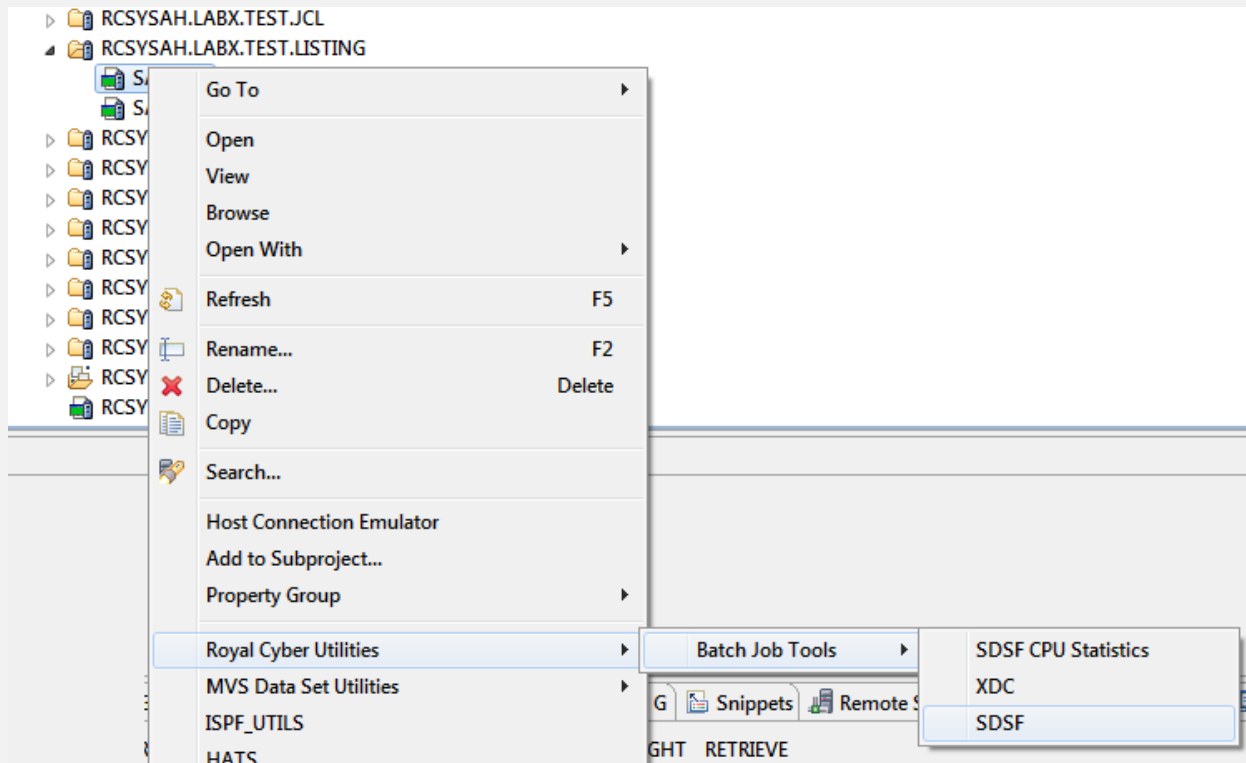
HATS/RCP driven from RDz Menu Manager

The HATS/RCP application - *now an RDz/Eclipse plugin* - is typically triggered from an RDz custom Context Menu. RDz Context Menus are easily modified/enhanced using Menu Manager (the free RDz scripting language described previously in this article).

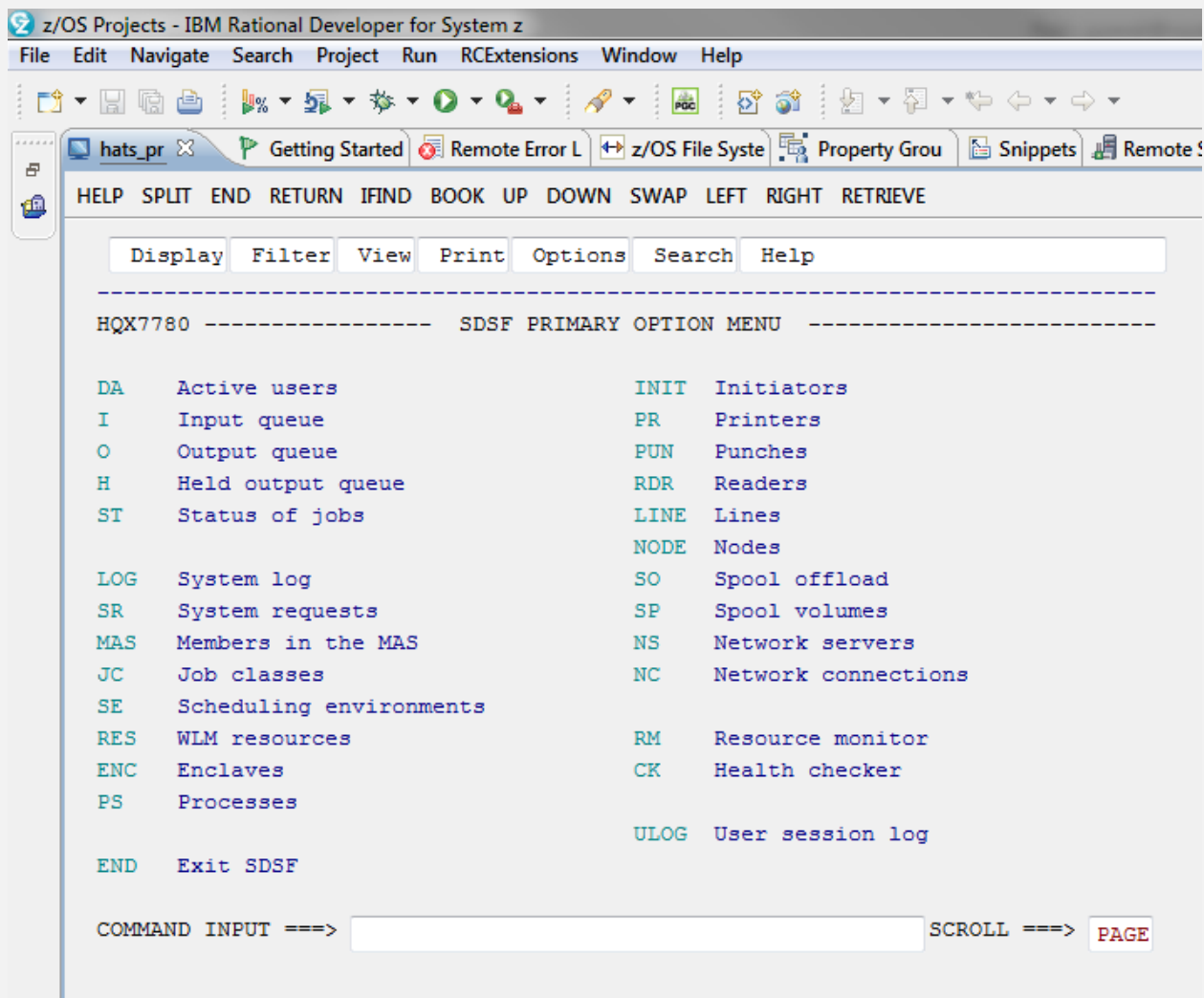
Variable and static content needed to launch the TSO/ISPF application captured through Context Menu dialogs is passed via parameters from the Context Menu to the HATS plugin. The plugin then runs a macro that accepts the parameters passed as input, and presents the TSO applications in an eclipse view.

Access to ISPF/SDSF Functionality from HATS/RCP and Menu Manager

From Figure 1, you can see that we have extended the RDz Remote Systems Explorer menu to include a customized sub-men set. This was done using Menu Manager. One of the options in the sub-menu is to access z/OS and present the SDSF Primary Option menu. After clicking the menu item, the HATS application opens in a new Eclipse view and presents the native SDSF interface to the user (Figure 2).

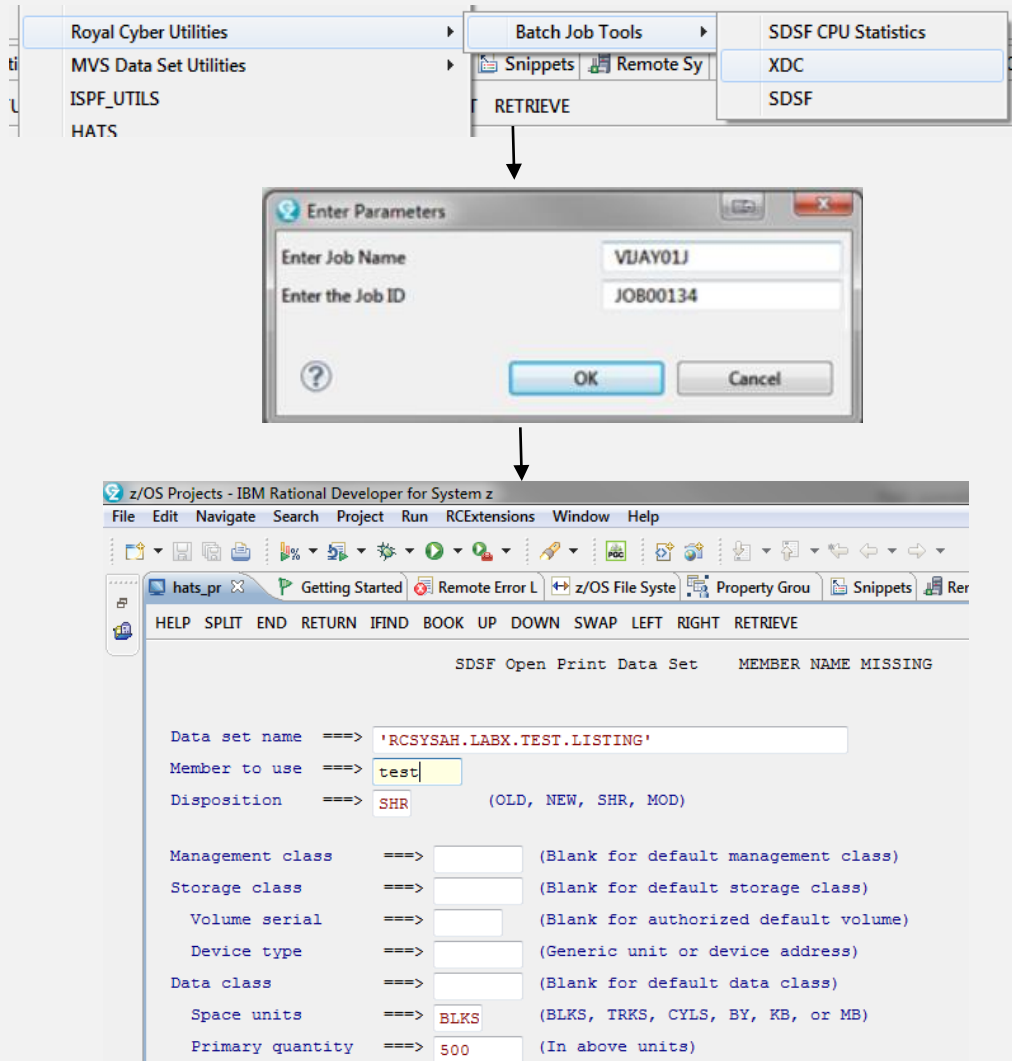


• Figure 1 – Menu Manager Context Menu Actions to SDSF from RDz



- Figure 2 – the SDSF Primary Option Menu rendered as an RDz View – screen-scraped using HATS/RCP

Figure 3 shows the same Context Menu accessing the SDSF XDC command directly. Because the XDC command requires a Job Name and Job ID, you would type in the Batch Job name and JES number. This is enough input for HATS/RCP to invoke the SDSF/XDC menu directly – both simplifying the workflow and integrating the process into the RDz workbench.



• Figure 3 – The SDSF XDC Command – activated from Menu Manager, passing Job Parameters from RDz to z/OS

Next steps...

While the HATS Toolkit is free, the time it takes to learn how to construct a HATS/RCP application and learn how integrate it with RDz through a custom Menu Manager driven enhancement to one or more RDz Context Menus is **not** free. HATS/RCP – Menu Manager development - while not rocket science is absolutely nontrivial (one of the IBM Client Technical Professionals knowledgeable in HATS/RCP estimated (conservatively) that it is a one-month learning curve). Few z/OS shops that we work with have qualified RDz development staff with that much free time.

So to both accelerate your learning, and to provide you with production-ready output (HATS/RCP functionality that you will integrate RDz with your z/OS functionality we've developed a services offering titled: [**"RDz-z/OS Integration using HATS/RCP and Menu Manager"**](#). This services offering is a one week workshop that combines hands-on learning with the construction of several production HATS/RCP functions. Your developers learn everything they need about HATS/RCP and Menu Manager while creating your first integration points from RDz to z/OS.

If you are interested in learning more about this offering please email: inquiries@royalcber.com

Juzer Ali has been working on Enterprise Modernization projects for the last 5 years. He has worked on various EM projects using HATS, RDz, and RD&T. He also works with the plug-in development team at Royal Cyber to develop plug-ins and utilities for RDz and other Rational products.



Juzer likes to follow European soccer and to develop mobile applications.

Royal Cyber in the News

In February we delivered our first IBM RDz Distance Learning classes – to over 150 new RDz users world-wide. As a first-time transition from IBM, which had been running Distance Learning for six years there were some hiccups with the logistics, but the results were as you can see below.

By the time this Newsletter goes to print there will (likely) **not** be availability left in the March Distance Learning session, but you can sign up for April (through the end of 2014) sessions here: <http://royalcyber.com/royal-cyber-rdz-distance-learning-training-schedule/>

@Syed/Marketing guys – please add additional content in the above style on that rash of Webinars and conferences you're at (I keep getting emails about them).

Rational EM News

- For details on the Royal Cyber RDz Rollout and project/task usage modeling contact us: inquires@royalcyber.com
- To view our technical newsletter - Volume 1, January 2014 – including additional deep-dive articles on RDz and RAAi: <http://royalcyber.org/mn/index.html>
- @Syed – this is where I'd love to be able to add content on new RDz plugins and things
-

What people are saying about Royal Cyber's RDz instruction:

.N

The instructor was awesome! He obviously knew this tool inside and out and was talented at teaching others about it. His relaxed and approachable attitude encouraged much discussion and questions....which is always a huge benefit in any class in my opinion.

I found it to be an excellent class... especially in the two sessions per day environment, which allowed me to apply the session in my own shops environment and retain the practical knowledge.

The class worked out great; I really appreciated the format. The instructor was very knowledgeable and did a good job with the challenging task of lecturing via phone.

The Royal Cyber online assistants were great because they could keep a running chat going, answering questions without interrupting the instructor's talk.

RDz - z/OS Integration using HATS/RCP and Menu Manager

Abstract: HATS allows conversational CLISTs, REXX, custom in-house panels and other 3rd party ISPF panels to be seamlessly integrated with RDz. This course is designed for Java/ Eclipse developers to develop Eclipse RCP plugins that can be installed by RDz developers for accessing native 'green screen' functionality.

Pre-requisites: Intermediate to advanced Java programming skills

Length:

- Training setup and customization: 40 hours
- Training delivery: 8+ hours – depending on content and modules requested

Topics/Learning Modules:

- Introduction to HATS
 - HATS toolkit overview, features and basic usage
 - Developing HATS applications
- Working with HATS screen customizations/combinations and macros
 - Working with Screen customizations and combinations
 - Macros
 - Global Variables
 - Using Default Transformation
 - Using Customized Transformations
 - Using Custom Business Logic (optional)
 - Creating HATS project with multiple entry points
 - Reusing existing connections
- Creating Eclipse RCP projects
 - Creating and packaging HATS applications as Eclipse plugins
 - Deploying plugins to update sites
 - Rolling out and installing plugins
- Working with Menu Manager and creating Menu Manager actions for HATS/RCP
 - Creating Menu Manager actions for calling HATS applications
 - Passing parameters to HATS applications
- Creating HATS projects with multiple entry points via Menu Manager
- Integrating custom CLISTs, REXX scripts, SCM panels and other green screen functionality with HATS
- Building Menu Manager actions that invoke HATS/RCP applications
- Designing custom Menu Manager drop-demo / context menus