June 2011

Enterprise Modernization Using Rational Developer for IBM System z

Ron Townsend
IBM Systems Lab Services and Training
rtownsen@us.ibm.com

TABLE OF CONTENTS

Overview

1.1 Introduction

Organizations face demands for new products and services, including global support for outsourced operations and integration with business partner and customer applications. Typically, mainframe organizations have a wealth of robust, proven legacy code, often customized over many years that efficiently and effectively handle core business application functions. The mainframe is a more powerful business asset than ever. More and more organizations use the platform as a highly available and security-rich environment for running core applications and they are extending existing applications and innovating workloads.

Those core processes probably contain millions of lines of code that provide valuable business intelligence which allows the management and processing of the majority of customer, product supply chain and other critical business data. The applications are typically based on IBM middleware software products such as CICS®, COBOL, PL/I, DB2®, and/or IMS™.

1.2 Why Enterprise Modernization?

Cost effective modernization efforts can provide deep insights of business applications, so embedded business rules can be identified, restructured and dead code removed and the impacts for code changes can be understood. Modernization can address this and the complex dimensions of architectural challenges, including fragmented business processes, workflows, data and tightly coupled application architectures. Understanding applications can improve the productivity of IT staffs and reduce maintenance costs through more automation and by eliminating the need to research, catalog and assemble information for each service request individually. IBM software helps organizations understand their applications as well as design, create, assemble, test and deploy high-quality Web, Web 2.0, Web services, portal and SOA applications for CICS, IMS and WebSphere® Application Server. By working with existing applications across platforms, companies can reduce time-to-market, improve business alignment for growth, cut costs and limit business risk.

A time-tested approach to leverage an organization's existing systems is to reuse rather than rewrite those systems. Once modernized, these IT systems offer greater agility, flexibility and robustness, and can boost productivity, enabling organizations to more quickly respond to marketplace dynamics and changing business needs and challenges.

Traditionally, organizations have managed mainframe development separately from other platform development with disconnected processes and tools. This separation not only hinders collaboration and productivity across the software lifecycle, but also leads to application failure or downtime. Supporting these applications requires an enterprise-scale approach to synchronizing software development, management, quality and release activities. IBM software for process, quality, change and release management helps

automate development processes, improve application quality and enhance team collaboration across multiple operating platforms throughout the application lifecycle.

IBM Enterprise Modernization solutions, delivered by the IBM Rational® Software Delivery Platform, help clients effectively and incrementally evolve core IT systems towards modern architectures and technologies, reducing maintenance burden and freeing up more resources to develop new business requirements and capabilities. This can help enable an organization to effectively create a new modernized and enhanced application in a fraction of the time and a fraction of the cost of total replacement or rewriting of applications.

IBM recognizes that organizations know their business requirements, staffing constraints and budget limitations better than anyone else. As a result, IBM can help assist organizations create a customized modernization roadmap and provide the tools needed to cost-effectively and incrementally evolve IT systems toward new architectures and technologies for nearly all platforms.

2 Rational Developer for System z

2.1 History

The Rational Developer for System z (RDz) product was introduced in 2003. The IBM Corporation's z/OS® Eclipse-based software development integrated development environment (IDE) for the 21st century and in that time, RDz has made the professional lives of z/OS programmers doing traditional application maintenance as well as "enterprise modernization" simpler and better.

The RDz tool does this by streamlining and refactoring z/OS development processes into a structured, efficient analysis, editing and testing operations, sustained by modern graphical user interface (GUI) tools, wizards and menus – making it perfect for new-to the-mainframe twenty and thirty-something developers.

The RDz environment also empowers veteran time sharing option (TSO) developers through faithful interactive system productivity facility (ISPF)-emulation and exceptional tooling for tough everyday z/OS software maintenance and support tasks (data flow analysis, control flow analysis, etc.). Via its powerful, simple GUI tools the Rational Developer for IBM System z® tool has helped refurbished the skill sets of veteran TSO developers, growing them into effective, contributing project staff for today's application requirements.

In each release since 2003, the Rational Developer for System z solution has been delivering incremental benefits. Today's business climate demands development technology that provides functionality and coverage for modern requirements such as wizards for handling XML, for developing and testing web services and stored procedures, etc., while at the same time shortening the maintenance, support and development lifecycles. This is achieved by eliminating routine tasks, significantly lowering the number of keystrokes needed to work effectively.

In its current version, Rational Developer for System z (RDz) software can help organizations enhance their core development capabilities. It uses the power of the Eclipse platform, a powerful and extensible workstation-based development environment, combining tools for creating COBOL, PL/I, C/C++, Java™, or Assembler transactional applications in Customer Information Control System (CICS) and IBM Information Management System (IMS), Java Enterprise Edition for WebSphere, and Web 2.0 applications using EGL. Providing this multitude of tools in a single unified development environment allows organizations to combine various technologies in order to help differentiate their applications from competitors and offer rapid application development support to diverse enterprise application development teams. With comprehensive development tools to help create, deploy and maintain traditional transactional and modern composite applications, Rational Developer for System z software allows developers with different technical backgrounds to easily participate in important technology projects.

3 How Rational Developer for System z Aids in Enterprise Modernization

3.1 Improve z/OS application development using z/OS Integrated Development Environment to improve productivity

Organizations struggle to tap into their traditional IT professionals' decades of experience and domain knowledge to improve current enterprise applications and to take advantage of new architectures and technologies. At the same time, organizations try to make it easier for developers with new skills to learn and reuse existing application assets to create services and to deploy new workloads on existing infrastructures. Making the move to modern development environments greatly improves productivity. Modern user interfaces for z/OS developers can help reduce training costs because they are simpler to understand and because colleges teach Eclipse-based tooling. The Windows-like feel simplifies interaction with z/OS. Just point and click to allocate, copy and move z/OS files and data sets. This feature makes the product more attractive to younger developers.

Improve productivity with an integrated and multiplatform development environment Built on Eclipse open-source technology and written to CICS, IMS, DB2, and Java EE specifications, Rational Developer for System z software is designed to optimize and simplify application development for traditional processes as well as more modern application development. Employing the common services of an integrated development infrastructure, helps facilitate reuse, better manage and provides improved communication. It also helps reduce requirements for manual integration—ultimately driving a shorter and more efficient development process. Eclipse plug-in technology allows you to integrate complementary development tools to extend the functionality of the total platform so you can interoperate with other Eclipse technology-based products.

Workstation real-time syntax checking can help reduce host CPU usage. Fewer COBOL and PL/I program compiles would be required on z/OS since developers can catch their syntax errors on the workstation before compiling it on z/OS. Simplified help with language constructs also helps prevent coding errors.

3.2 Rational Developer for Systems z Capabilities for Application Development

Rational Developer for System z software has key capabilities for application development that assist Enterprise Modernization. Those capabilities include:

- CICS, IMS, DB2, Batch, z/OS, IBM WebSphere Application Server, WebSphere Compute Grid (Modern/Java Batch), development and maintenance tools.
- Real-time validation of code during code creation, including embedded language support for CICS and DB2.
- Integrated life-cycle development with IBM Rational Team Concert™ software, IBM Rational ClearCase® software, CA Endevor, and IBM Software Configuration and Library Management (SCLM) Developer Toolkit.
- Browse and update VSAM dataset content on System z using IBM File Manager integration or access problem reports for diagnosing mainframe application faults using IBM Fault Analyzer integration.
- End-to-end heterogeneous application debugging using IBM Debug Tool. Applications can be visually debugged from HTML hosted in WebSphere Application Server completely through to transactions executing in CICS or IMS, to data access and stored procedures in DB2.

4 Running a Simple COBOL Program Remotely Using Rational Developer for System z

4.1 Preparation

1) Open the Rational Developer for System z tool.
2) Switch to the z/OS Projects perspective:

   a) In the workbench, select Window > Open Perspective > Other. The Select Perspective wizard opens.

   b) Click z/OS Projects.

   c) Click OK. The z/OS Projects perspective opens.
3) Create a connection to a Remote system:

   a) In the Remote Systems view, expand NEW CONECTION and select z/OS.

   b) Right click and choose New Connection. The New Connection window opens.

   c) Select a profile name from the drop-down list.

   d) Enter the following values in the on this window:

      Host Name – Enter the TCP/IP address of the system you are connecting to.

      Connection Name – Enter the short name you want to call the system.

      For example, MYSYSTEM

      Description – Enter a description of your choice.

      Verify host name – Select this check box to verify that the host is valid before attempting to connect.

To define the connection using default values for the MVS™ Files, z/OS UNIX Files, and z/OS UNIX® Shells subsystems, click Finish. To set properties for these subsystems, click Next. The wizard opens a properties window for each subsystem. These pages display the properties of the underlying services used by each subsystem.

On the MVS Files page, you can select from the following server launchers. If you are not sure which option to choose, contact your system administrator.

Remote daemon: Establishes a connection using the remote daemon to start the server. To use this option, the remote daemon must be running on the remote system. If you choose this option, specify the following additional options:

Daemon Port: Specify a valid port number.

Authentication method: Choose a method for authenticating with the remote system. Select userid/password if you log on to the remote system using a user ID and password. Select certificate if you use client certificate authentication. Client

certificate authentication is for users who need to connect to a remote system using a device such as an integrated circuit card (like Smart Card).

REXEC: Establishes a connection using the REXEC service to start the server. To use this option, the REXEC service must be running on the remote system. If you choose this option, specify the following additional options:

Path to installed server on host: Specify a valid path command to where the server is installed on the remote system. You can specify a path that is relative to the directory where you run the REXEC command or the full path to the location where the server is installed. For example, dstore or /usr/bin/dstore.

Server launch command: For IBM eServer™ zSeries®, the command is server.zseries.

Port: Specify a valid port number.

Auto-detect SSL: Automatically detects if SSL is running on the server and connects using SSL.

Use SSL for network communications: Connects using SSL.

Connect to running server: Establishes a connection with a server that is already running on some known port. To use this option, the server needs to be started before you attempt to define a connection in the Remote System Explorer. The port must be specified on the Subsystem properties page before you can connect to the server. If you choose this option, you also need to specify the Use SSL for network communications option, which connects using SSL

SSH: Establishes a connection using secure shell support. To use this option, the SSH service must be running on the remote system. If you choose this option, specify a valid path command to where the server is installed on the remote system. You must also specify a server launch command. For zSeries, the command is server.zseries. Select Password authentication authentication if you log on to the server using a password or key authentication if you log on using a private/public key pair to authenticate with the server. If you choose key authentication and the key pair does not exist, the required key pair is automatically generated and exchanged with the remote SSH server for future requests. You are prompted for a password to the remote system to enable this exchange. Any subsequent requests to the remote system through SSH using key authentication do not require a password.

Click Finish. The Remote Systems view displays the short name of the new connection with five nodes under the connection name:

z/OS UNIX Files is the z/OS UNIX file subsystem. This node contains two folders: My home and Root. You can create additional z/OS UNIX file folders by adding new filters to this node.

z/OS UNIX Shells is a command subsystem. When you open a z/OS UNIX command shell, its name appears under this node.

MVS Files is the MVS file subsystem. This node contains three folders: My Data Sets displays MVS files that match the filter *userid*.* in which *userid* is the user ID with which you connected to the remote system. You can create additional MVS file

folders by adding new filters to this node. You can change the sort order of data sets by using the MVS Files preference page. Retrieved Data Set displays data set names searched for and added by using the Retrieve Data Sets action. My Search Queries displays search queries you have run and saved in the Remote z/OS Search view.

TSO Commands is a command subsystem. When you open a TSO command shell, its name appears under this node.

JES is the JES subsystem. This node contains two folders: My Jobs displays jobs submitted under the user ID with which you connected to the remote system. You can create additional job folders by adding new filters to this node. Retrieved Jobs displays jobs searched for and added by using the Retrieve Job action.

4) Allocate partitioned datasets on the remote system. Use the steps described below this list of dataset names to allocate each partitioned dataset using these characteristics. In the following partitioned dataset names, <*HLQ*> is the user ID you used to log on to the remote system.

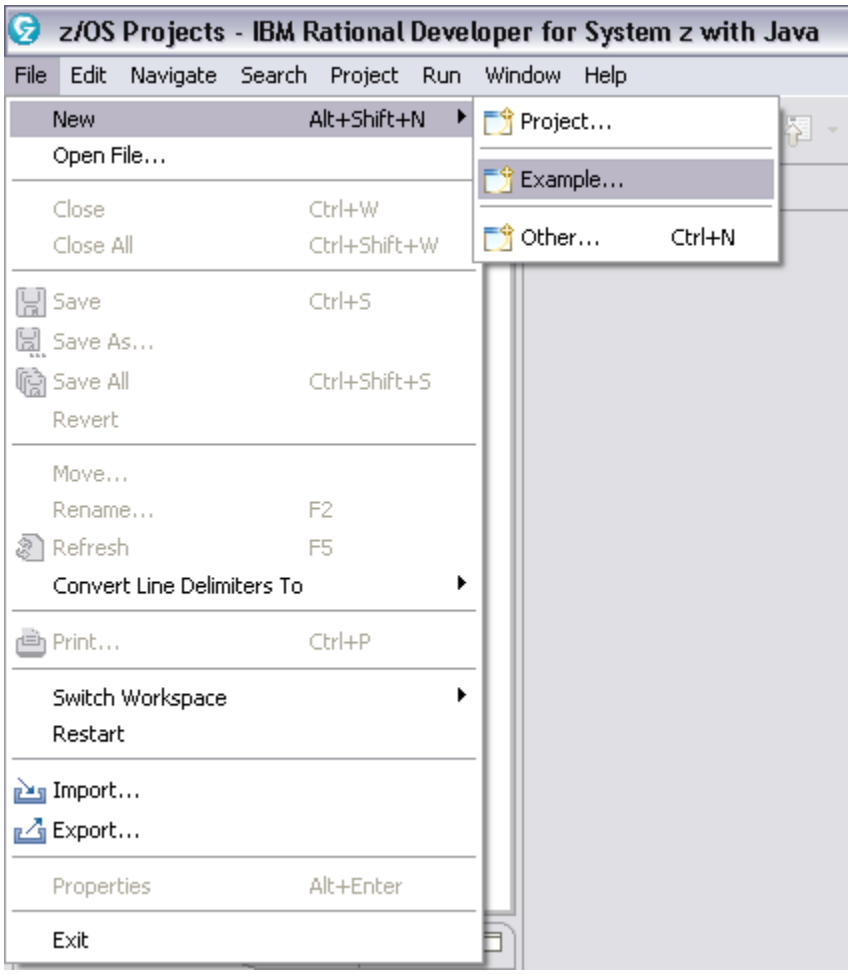
Data set name	Category	Type
<*HLQ*>.REMOTE.COBOL	SOURCE	COBOL
<*HLQ*>.COBOL.OUTPUT*	LISTING	COBOL
<*HLQ*>.COBOL.SYSDEBUG*	LISTING	COBOL
<*HLQ*>.COBOBJS.OBJ*	SOURCE	COBOL
<*HLQ*>.COBOL.COPYLIB*	SOURCE	COBOL

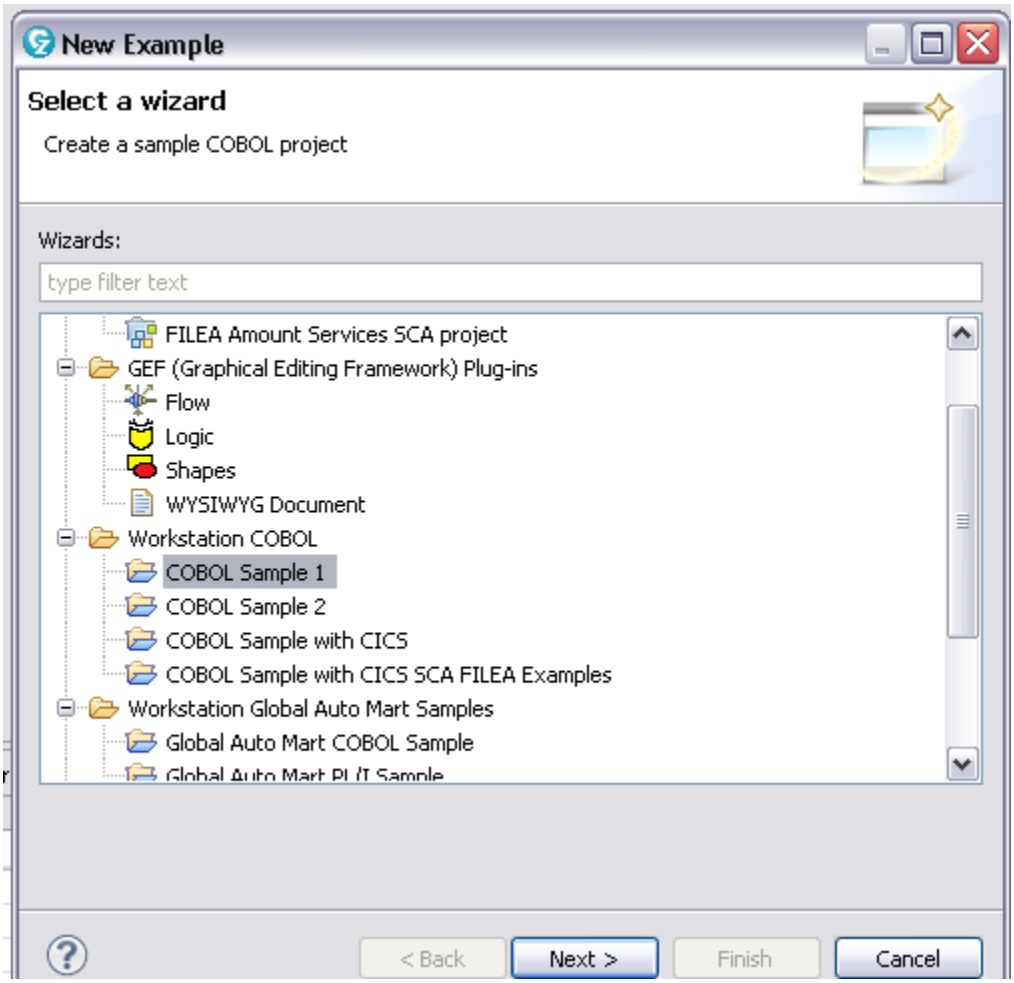
For each of these data sets, do these steps:

a. Select the MVS Files subsystem in the Remote System view and click New > Allocate Partitioned Data Set.
b. On the Allocate Partitioned Data Set page, do these tasks:
    i. Accept the name that is provided in the Host Short Name field.
    ii. Select the high-level qualifier for the data set from the drop-down list.
    iii. In the Data Set Name field, specify the name that follows the combination of high-level qualifier and period. For example, if the high-level qualifier is MYFILES and if you want to allocate the data set MYFILES.REMOTE.COBOL, specify the string REMOTE.COBOL in the Data Set Name field.
    iv. Click Next.
c. On the Data Set Allocation page, select the Specify characteristics by usage type radio button and then select the corresponding value from each list box as shown in the table.
d. Click Finish.

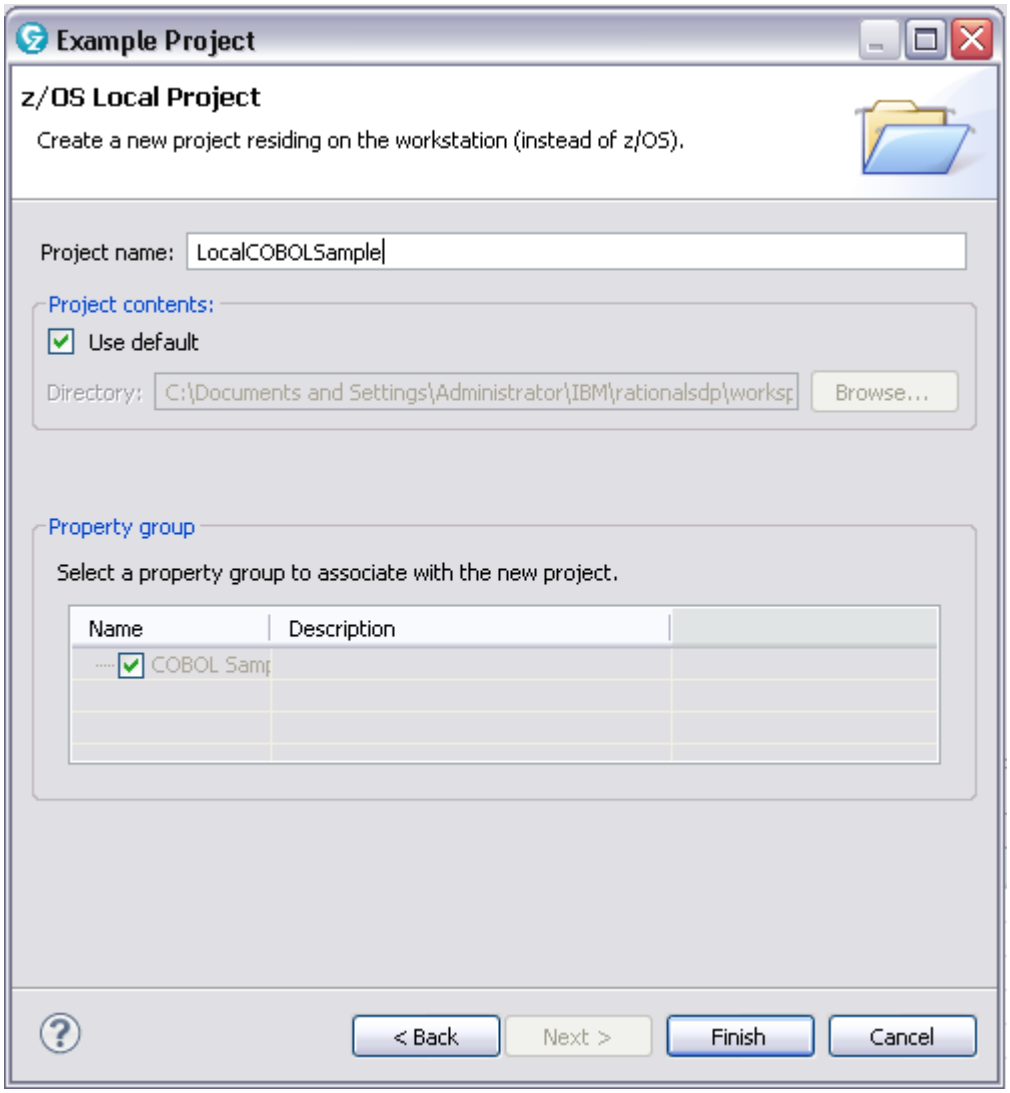
4.2 Obtain Sample COBOL Code for Exercise

a) Click File > New > Example. Expand Workstation COBOL and Click COBOL Sample 1 in the list. Click Next

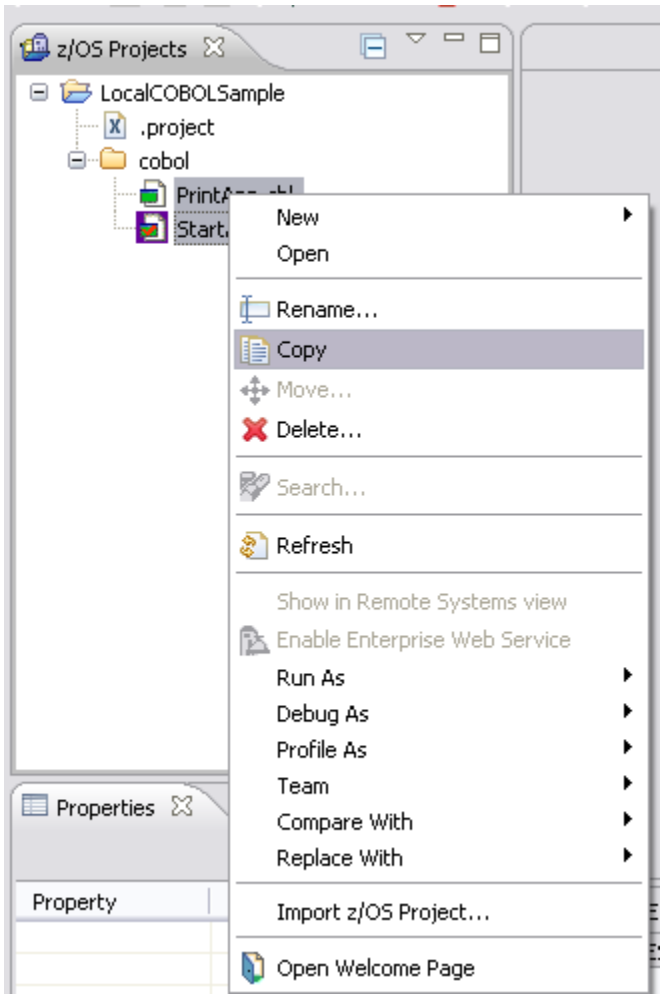


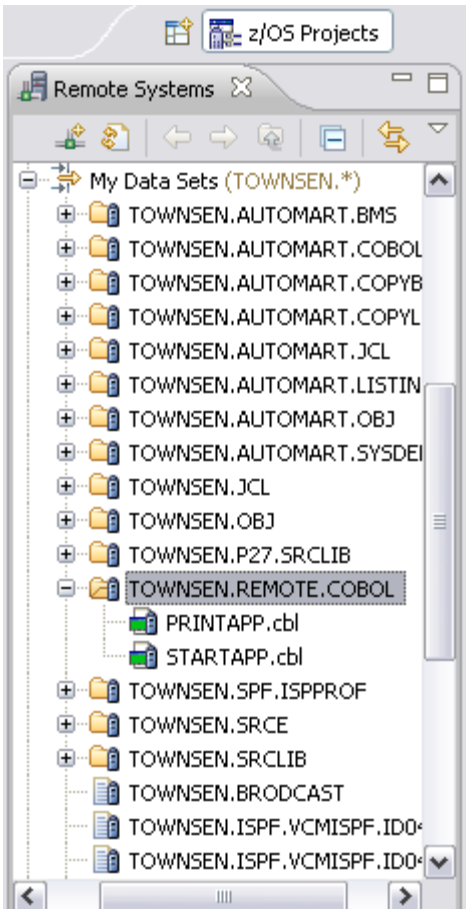

New Example
Select a wizard
Create a sample COBOL project
Wizards:
type filter text
FILEA Amount Services SCA project
GEF (Graphical Editing Framework) Plug-ins
Flow
Logic
Shapes
WYSIWYG Document
Workstation COBOL
COBOL Sample 1
COBOL Sample 2
COBOL Sample with CICS
COBOL Sample with CICS SCA FILEA Examples
Workstation Global Auto Mart Samples
Global Auto Mart COBOL Sample
< Back
Next >
Finish
Cancel

b) Specify LocalCOBOLSample as the project name and click Finish.



c) Copy the COBOL source files, PrintApp.cbl and StartApp.cbl, to the <*HLQ*>.REMOTE.COBOL partitioned data set.



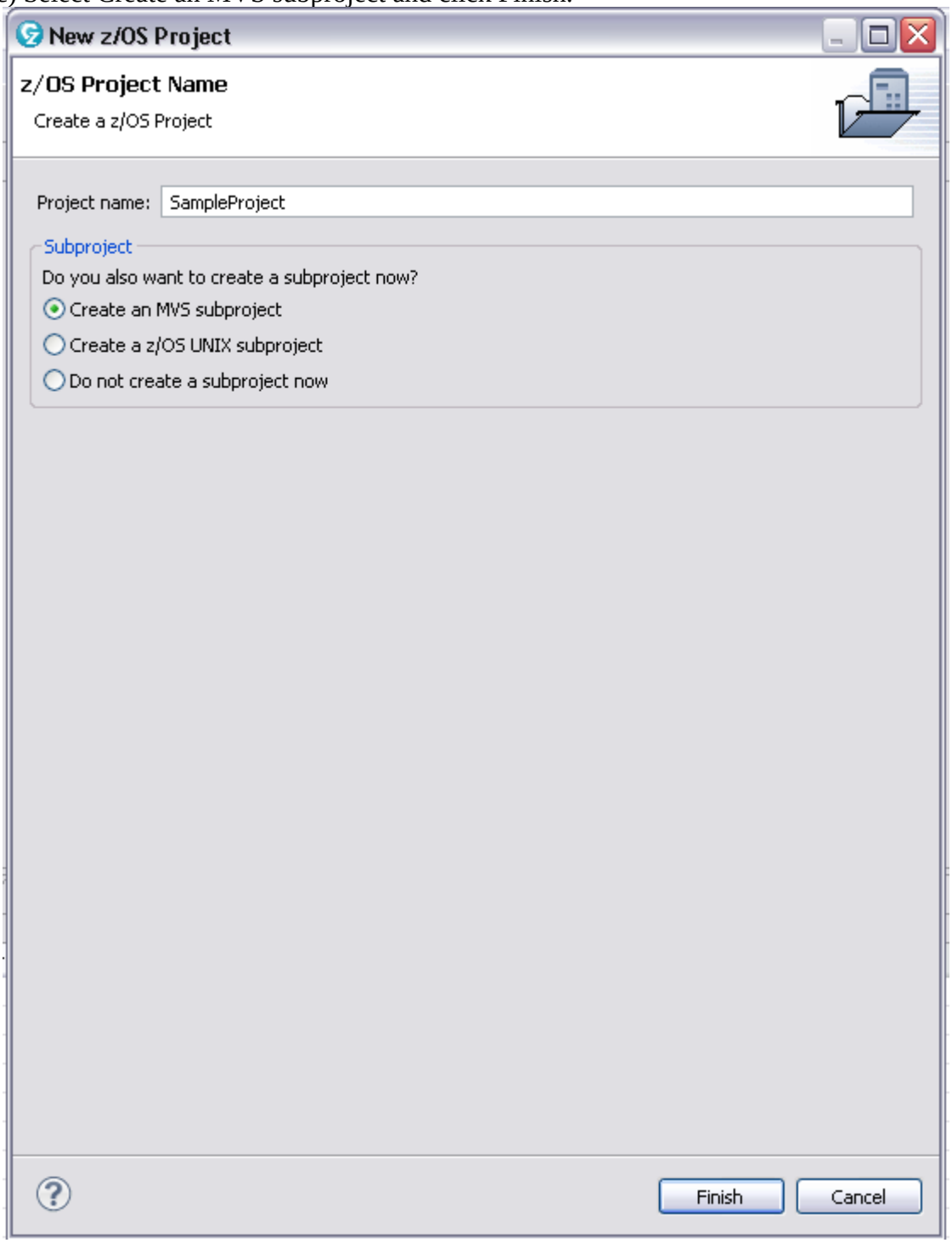

z/OS Projects
Remote Systems
My Data Sets (TOWNSEN.*)
TOWNSEN.AUTOMART.BMS
TOWNSEN.AUTOMART.COBOL
TOWNSEN.AUTOMART.COPYB
TOWNSEN.AUTOMART.COPYL
TOWNSEN.AUTOMART.JCL
TOWNSEN.AUTOMART.LISTIN
TOWNSEN.AUTOMART.OBJ
TOWNSEN.AUTOMART.SYSDEI
TOWNSEN.JCL
TOWNSEN.OBJ
TOWNSEN.P27.SRCLIB
TOWNSEN.REMOTE.COBOL
PRINTAPP.cbl
STARTAPP.cbl
TOWNSEN.SPF.ISPPROF
TOWNSEN.SRCE
TOWNSEN.SRCLIB
TOWNSEN.BRODCAST
TOWNSEN.ISPF.VCMISPF.ID0
TOWNSEN.ISPF.VCMISPF.ID0

4.3 Create a z/OS Project and MVS Subproject

a) In the z/OS Projects view, click New > z/OS Project. The New z/OS Project wizard opens.

b) In the Project name field, type SampleProject.

c) Select Create an MVS subproject and click Finish.



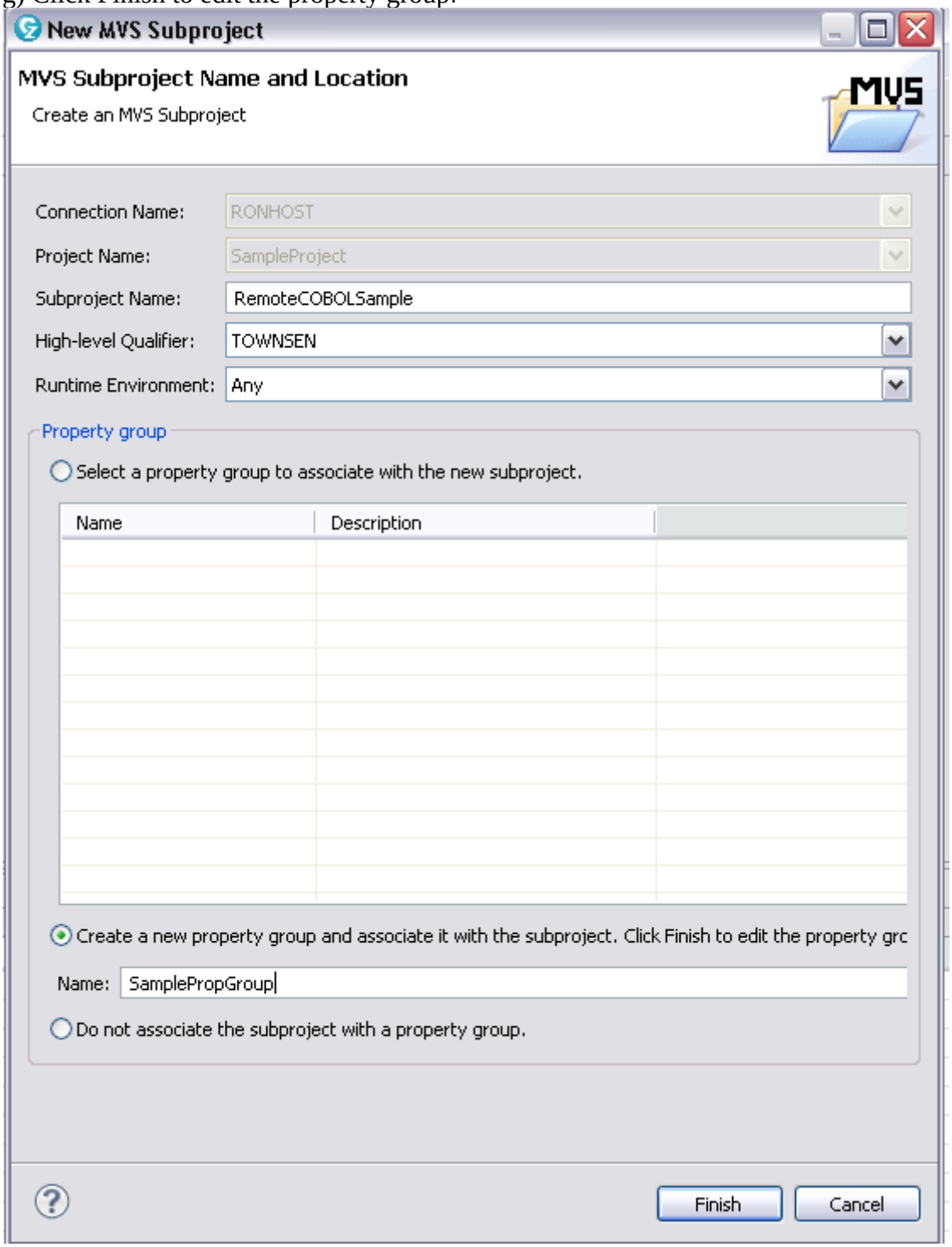
d) In the Subproject Name field, type RemoteCOBOLSample. The High-level

Qualifier defaults to the user ID with which you logged on to the remote system.

e) In the Property Group area of the New MVS Subproject window, click Create a new property group and associate it with the subproject.

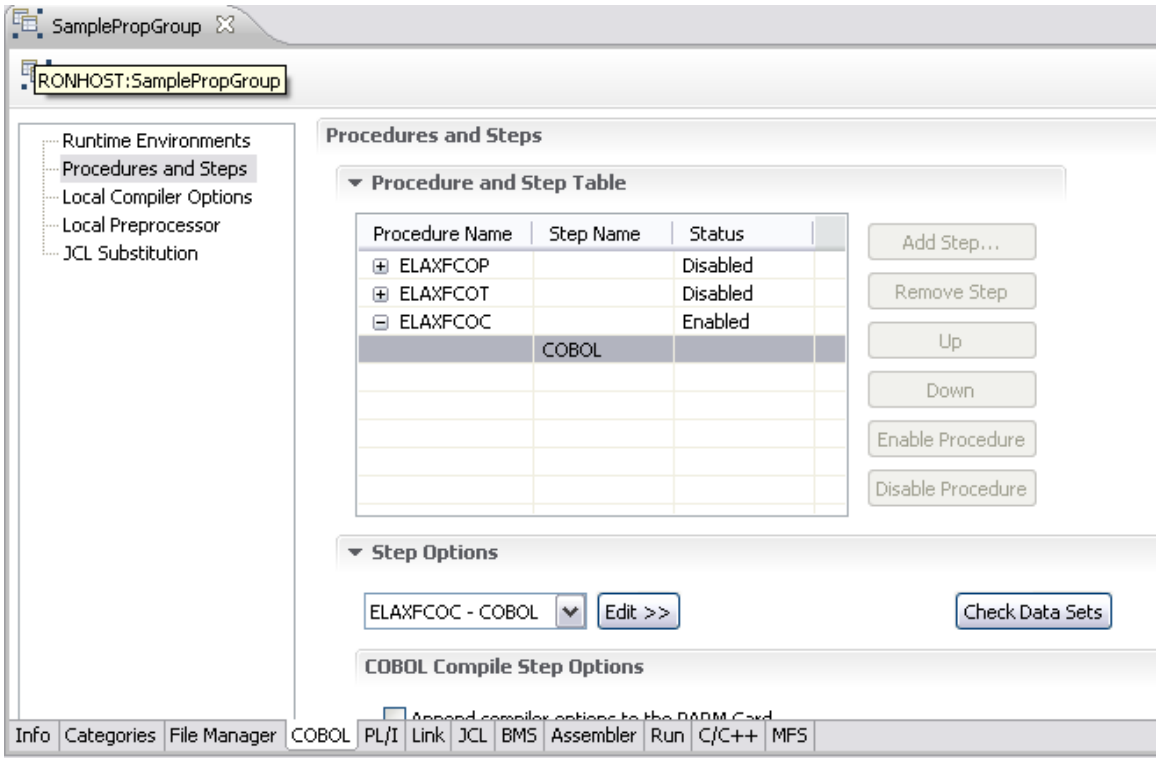
f) Name the property group SamplePropGroup.

g) Click Finish to edit the property group.



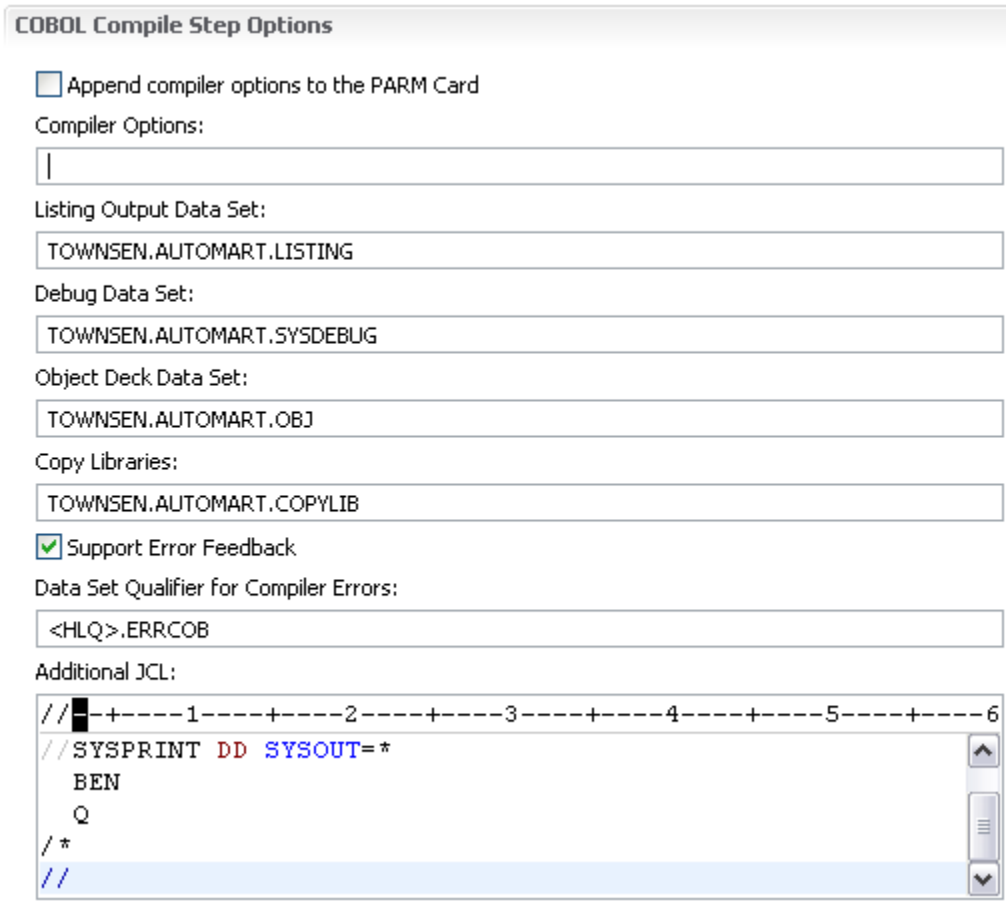
To set build options in the new property group:

a) In the property group editor, click the COBOL tab and then click Procedures and Steps.
b) Expand the ELAXFCOC procedure and double-click COBOL to edit the COBOL step.

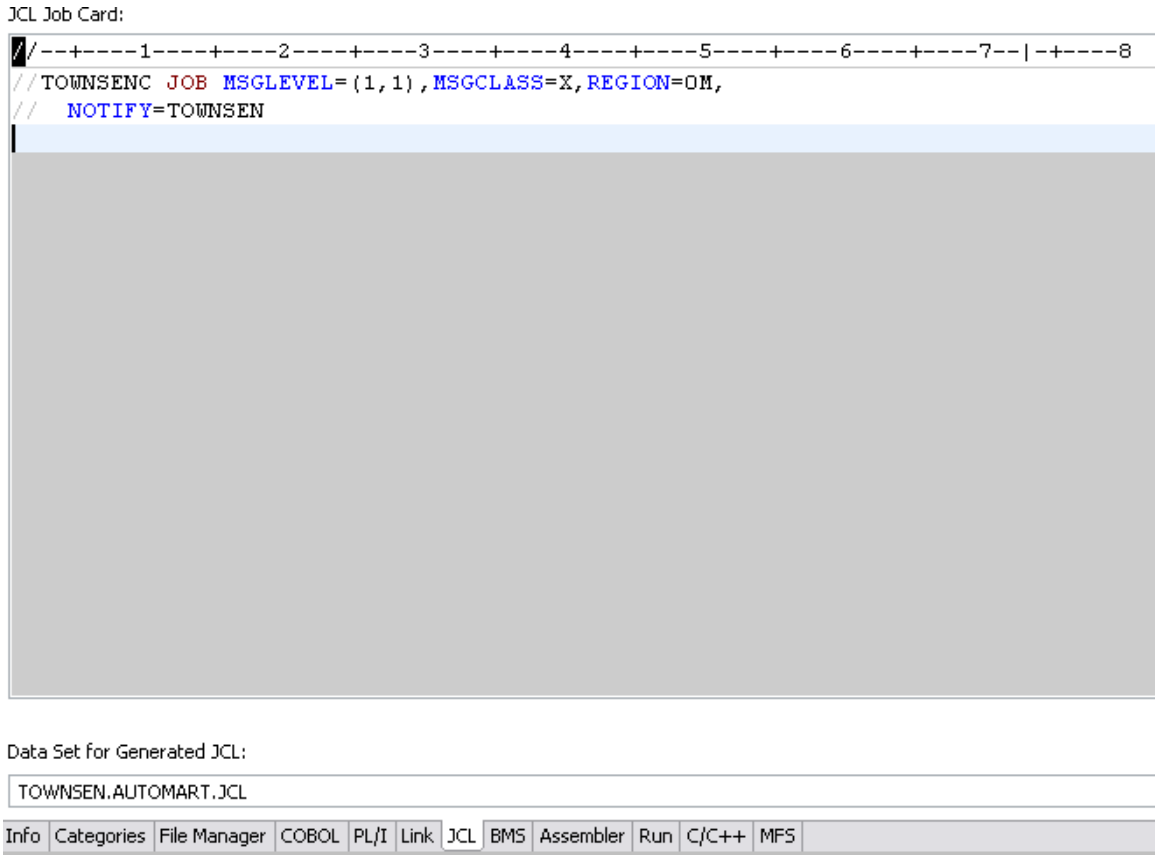


c) Copy and paste the following JCL into the Additional JCL text box:

```
//SYSOUT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
  Ben
  Q
/*
//
```


COBOL Compile Step Options
Append compiler options to the PARM Card
Compiler Options:
Listing Output Data Set:
TOWNSEN.AUTOMART.LISTING
Debug Data Set:
TOWNSEN.AUTOMART.SYSDEBUG
Object Deck Data Set:
TOWNSEN.AUTOMART.OBJ
Copy Libraries:
TOWNSEN.AUTOMART.COPYLIB
Support Error Feedback
Data Set Qualifier for Compiler Errors:
<HLQ>.ERRCOB
Additional JCL:
//-+----1----+----2----+----3----+----4----+----5----+----6
//SYSPRINT DD SYSOUT=*
BEN
Q
/*
//

d) Click the JCL tab in the property group editor and ensure that the default JOB card includes a PROCS statement that points to the procedure library on the remote system. If you do not know the location of the procedure library, ask your system administration.



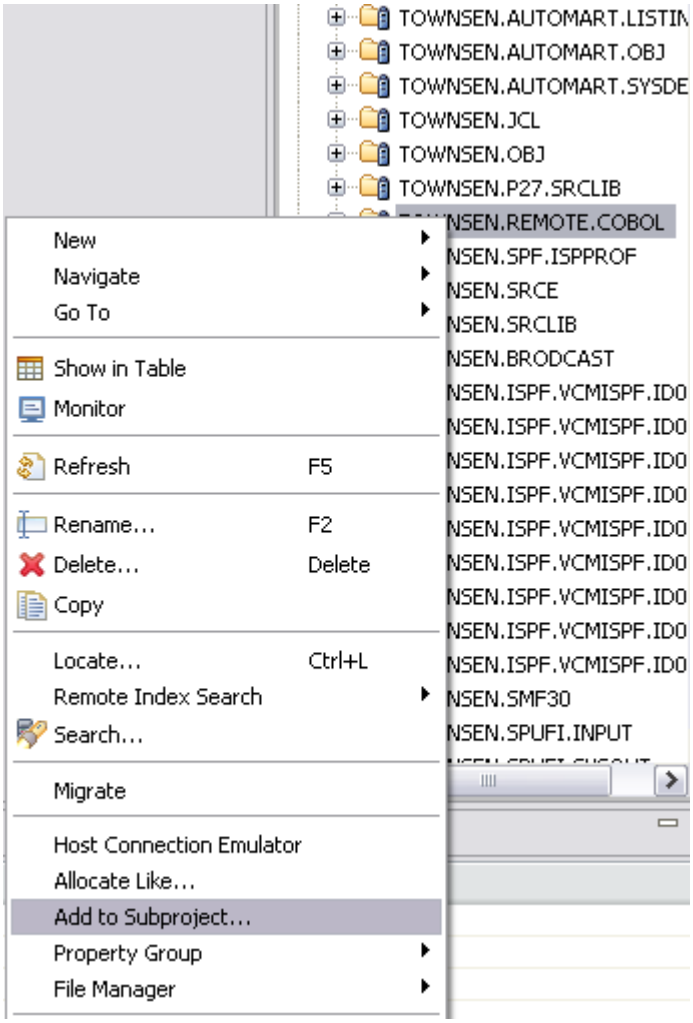
e) Close and Save the new property group.

4.4 Add Partitioned Dataset to the RemoteCOBOLSample Subproject and Run the Application

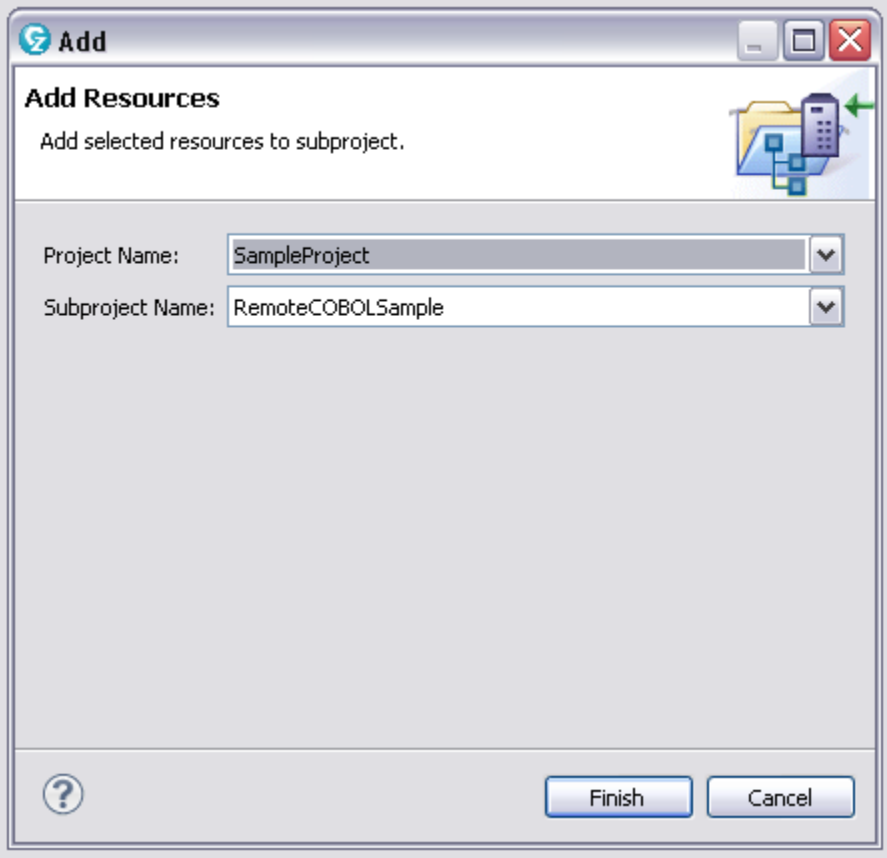
Add the <*HLQ*>.REMOTE.COBOL partitioned data set to the RemoteCOBOLSample subproject:

1) Select the data set in the Remote Systems view.

2) Right-click and select Add to Subproject.

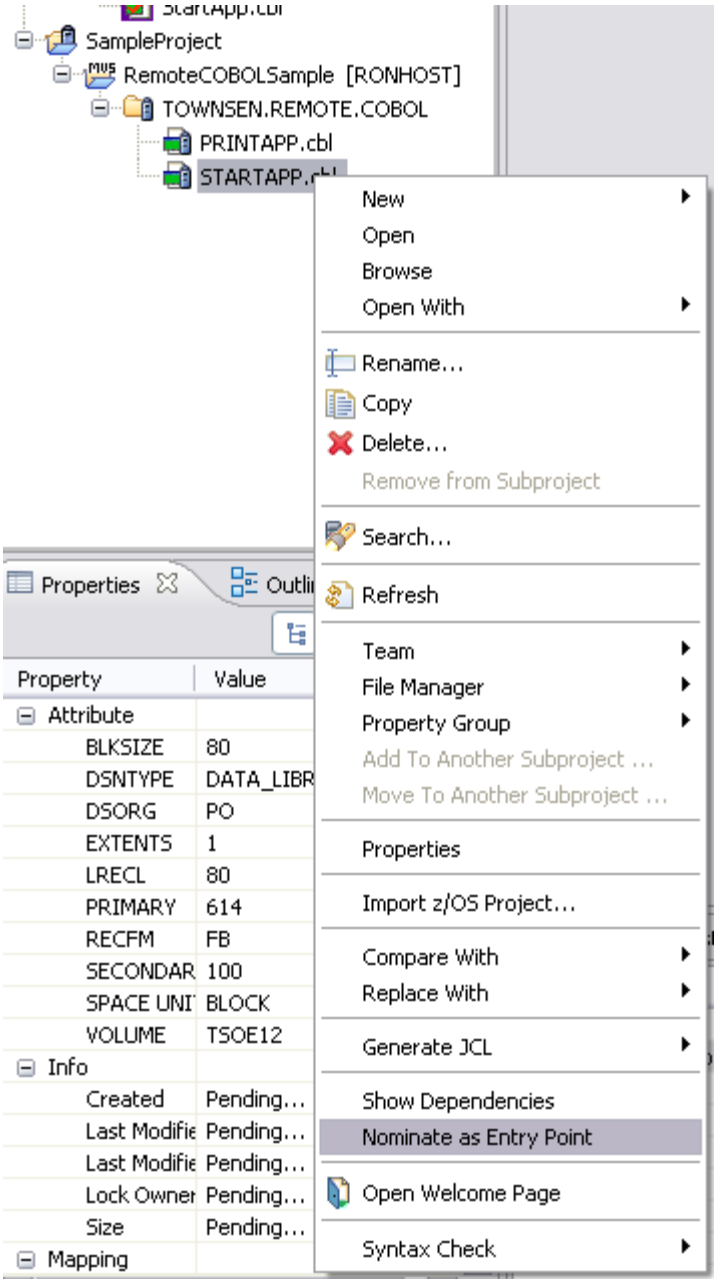


3) Type or select the subproject name and click Finish.

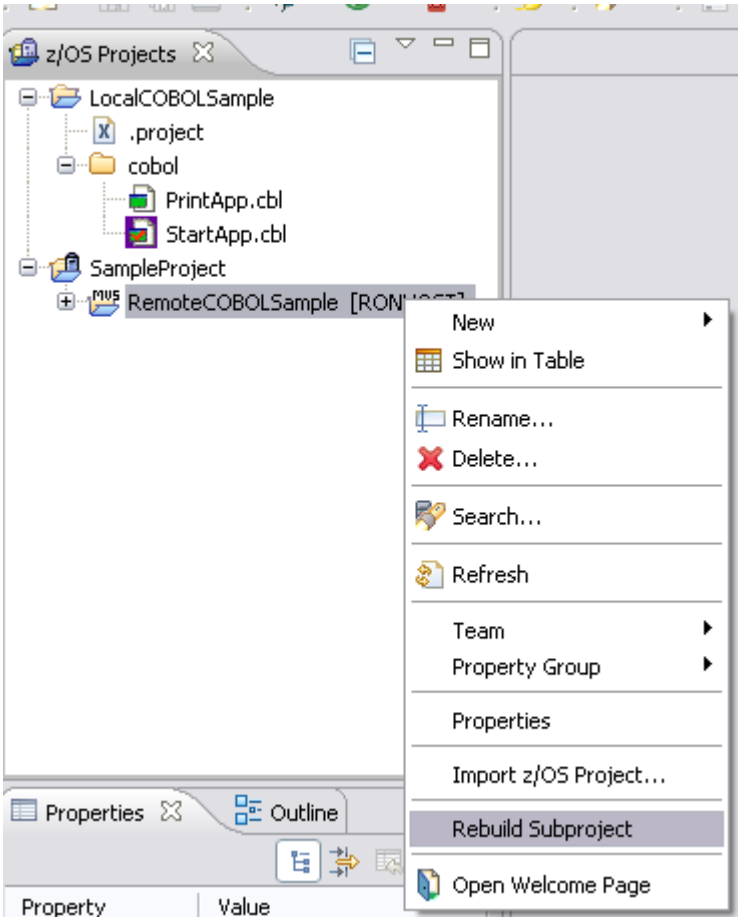

Add
Add Resources
Add selected resources to subproject.
Project Name:
SampleProject
Subproject Name:
RemoteCOBOLSample
Finish
Cancel

Build and run the application:

1) In the RemoteCOBOLSample subproject, select STARTAPP.cbl and click Nominate as Entry Point.



2) Select RemoteCOBOLSample and then click Rebuild Subproject. At the end of the build operation, the listings, object decks, and generated load module are added to the subproject.



3) Select the generated load module, and click Run > Run to run the application.

5 For More Information

- To learn more about IBM Enterprise Modernization solutions for System z, please contact your IBM sales representative or IBM Business Partner, or visit: ibm.com/rational/modernization

- For the executive summary of an independent analyst's perspective of both Rational Developer for System z and IBM Problem Determination Tools, visit: ftp://ftp.software.ibm.com/software/htp/pdtools/PD_Tools_ES_1st-ed_Jan09.pdf

- To see demos of RDz with PD Tools, visit: http://rational.dfw.ibm.com/atdemo/atdemo_rdz_zosad_recorded.html

- To quickly try practical scenarios guided by self-paced exercises 24x7 with our free System z sandbox, visit: ibm.com/developerworks/downloads/emsandbox

- For RDz Product details, visit: http://publib.boulder.ibm.com/infocenter/ratdevz/v8r0/index.jsp

- For Videos of many RDz functions, visit: http://publib.boulder.ibm.com/infocenter/ieduasst/rtnv1r0/index.jsp?topic=/com.ibm.iea.rdz/rdz/wdz76.html

IBM Systems and Technology Group
Route 100
Somers, New York 10589
U.S.A.
Produced in the United States of America,
06/2011

All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

Performance is in Internal Throughput Rate (ITR) ratio based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput 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 improvements equivalent to the performance ratios
stated here.

ZSW03190-USEN-00