



Solutions for Access and Integration to HP OpenVMS Applications

Jim Jennis, Integration Architect, WRQ

Manoj Winfred, Senior Consultant, WRQ

WRQ and HP

Pioneers, Partners & Leaders in World Class Computing Solutions

- WRQ—Over 22 years of connecting host applications with emerging technologies
- Multiple options for extending the life of existing OpenVMS & HP applications
 - Host access/emulation
 - Web-enablement
 - Composite applications
 - Itanium® migration
- WRQ is an HP pioneer, and has been a trusted HP partner for over 20 years
- WRQ has more than 10,000 customers on HP platforms worldwide

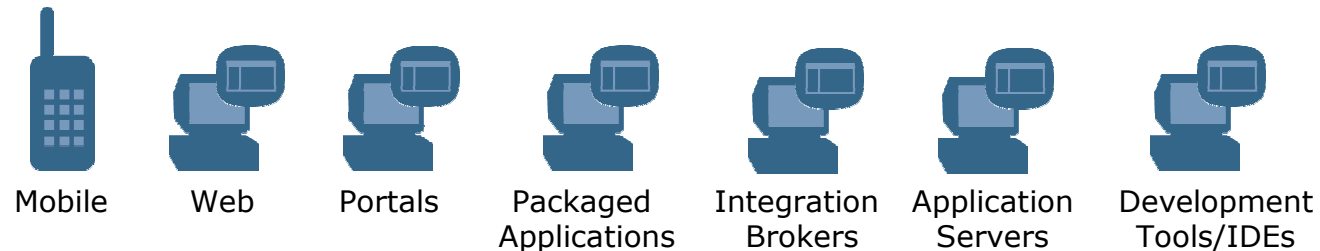
WRQ Value to HP Customers

- Extend/Enhance use of HP platforms
 - OpenVMS
 - HP-UX
 - Linux
 - HP e3000
- Integrate with third party platforms
 - IBM 3720/5250, OS/390, AS/400
 - UNIX—AIX, Solaris, etc
- A single vendor for legacy extension and integration
 - Terminal emulation — fat & thin
 - Web-refacing of legacy applications
 - Creation of web-services and composite applications from legacy functions and data

SOA Enables Enterprise Legacy Applications

Target Applications & Tools

Reuse legacy functions in new ways



Interfaces

.NET, COM, Java, EJB, Web-services, HTTP/XML, JMS

Composite Services

Combine components into high value services

Components

Represent business functions or data elements

Adapters

Abstracts host logic & data into components

OpenVMS specific adapters

Screen: Any VT-based application (VAX, Alpha, Itanium) including: All-in-1, FMS, FMS/Datatrieve, Cognos Powerhouse, 3GLs
Data: RMS Files, RMS/Cobol, RMS/Datatrieve, Oracle/RDB, CDD, Oracle

OpenVMS & UNIX

Mainframe

HP e3000

AS/400

Databases

Custom/
Packaged
Apps

Repository
Access & store components & services

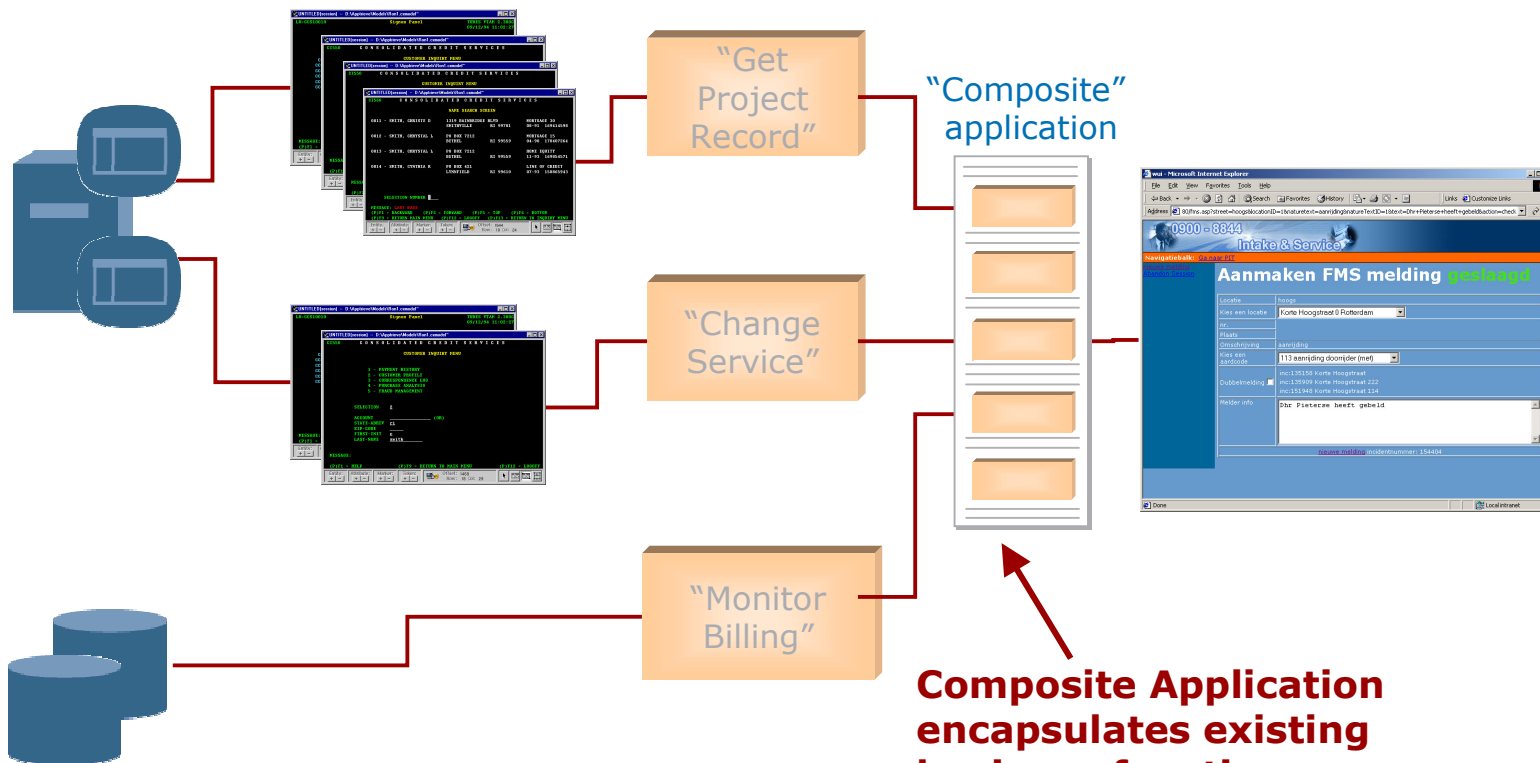
Composite Application Architecture

Existing
host applications
& databases

User steps to
complete a
business function

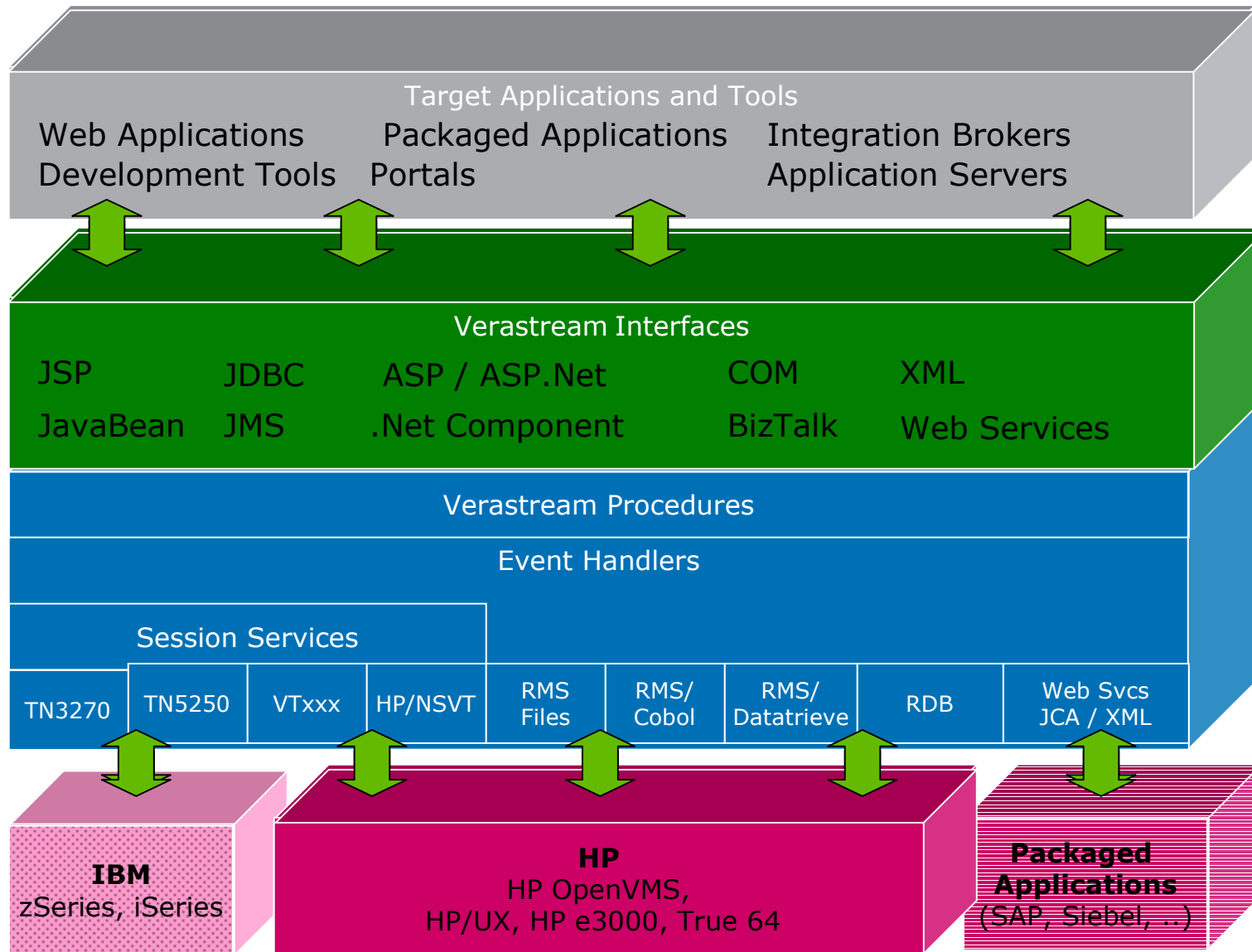
Reusable
business
components

New call center
web interface

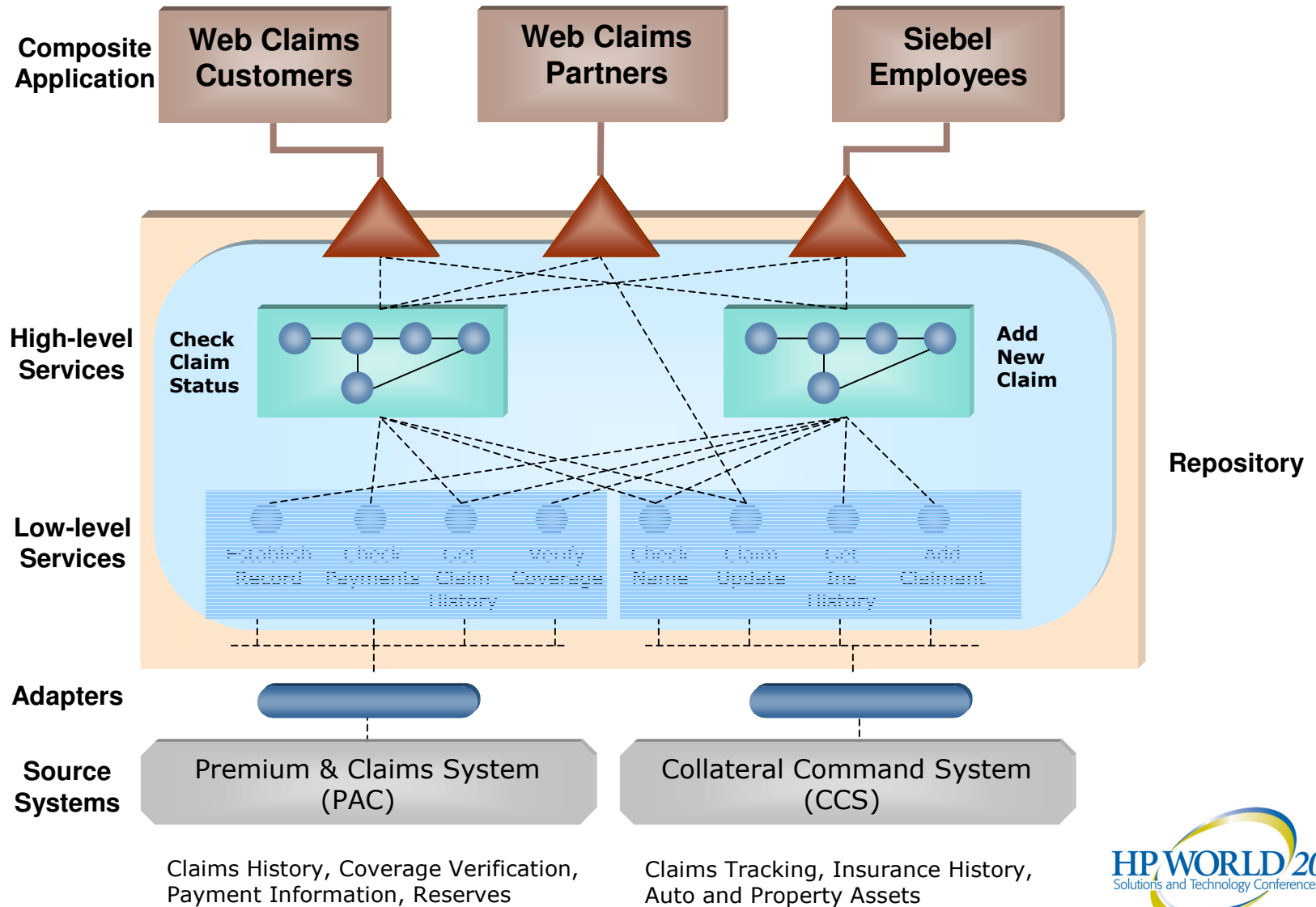


**Composite Application
encapsulates existing
business functions
allowing reuse and
helping accelerate new
application development**

Verastream Architecture



Verastream Composite Applications – Customer Example



Verastream Composite Application Tutorial

Freight Claims Payment—Verastream Demo

Combines/reuses:

- Data Layer →
 - RMS Data used/maintained by existing OpenVMS Cobol Applications
- U.I. →
 - FMS/Fortran Character Based Checking Account Application wrapped with Verastream Host Integrator
- API →
 - A legacy OpenVMS application written in C and wrapped with HP Bridgeworks to create a Java callable interface
- WEB →
 - A modern easy-to-use WEB user interface developed using Java/JSP

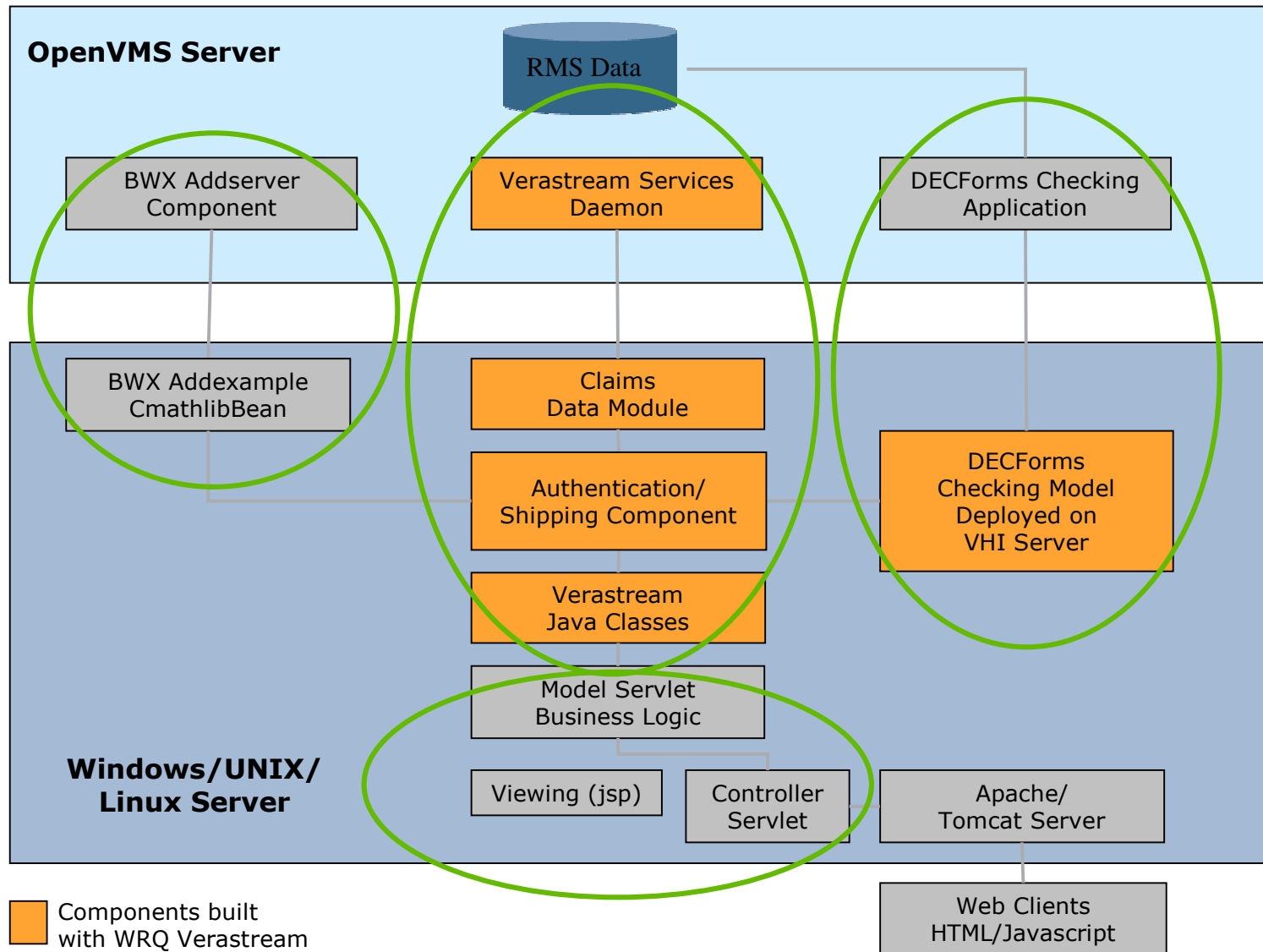
Verastream Composite Application Development Process

Freight Claims Payment Development Process

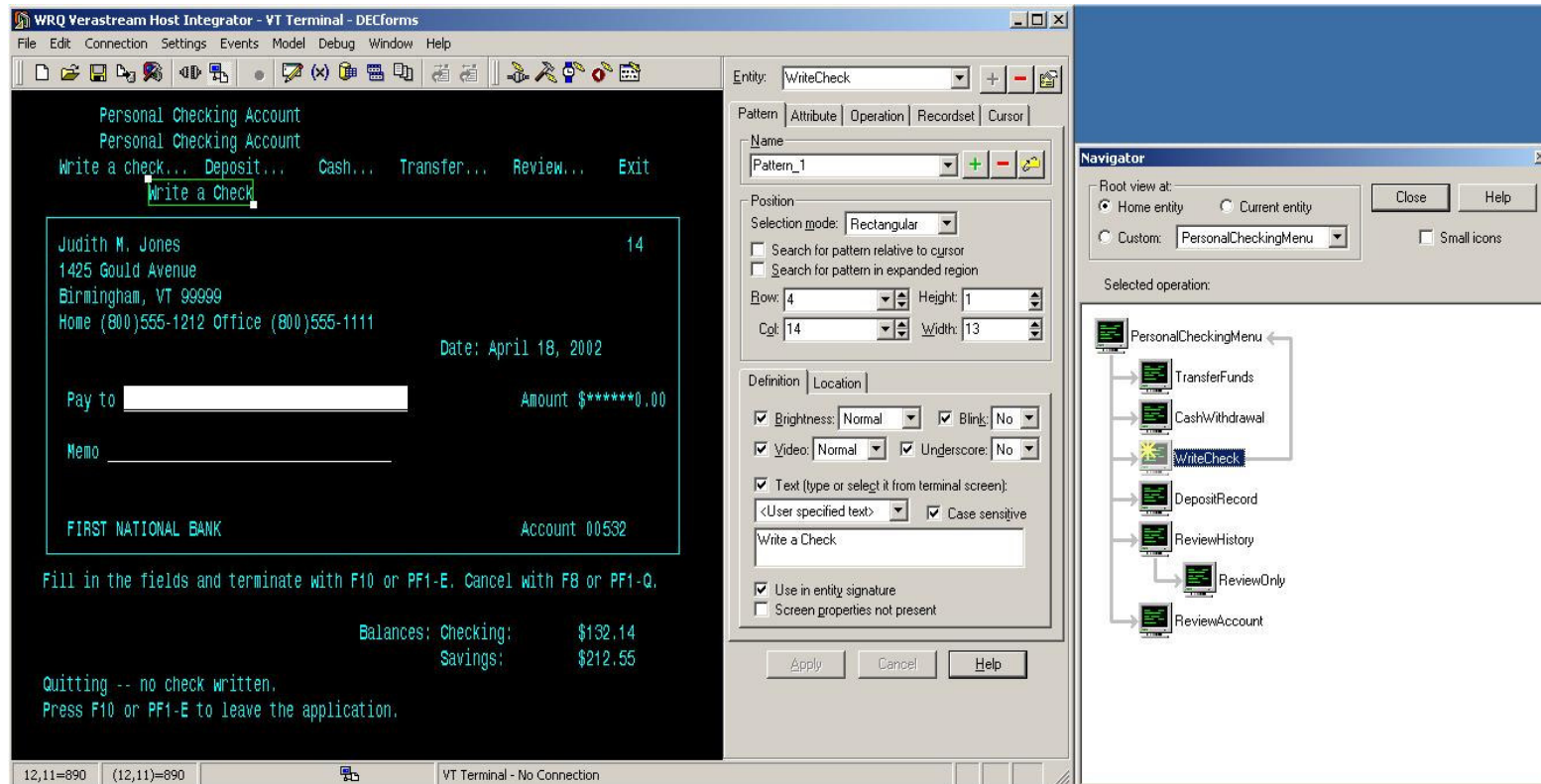
Five simple steps:

- Build the components
- Add the components to the Repository
- Assemble the composite application
- Deploy the composite application
- Test the composite application
- Mission Accomplished!

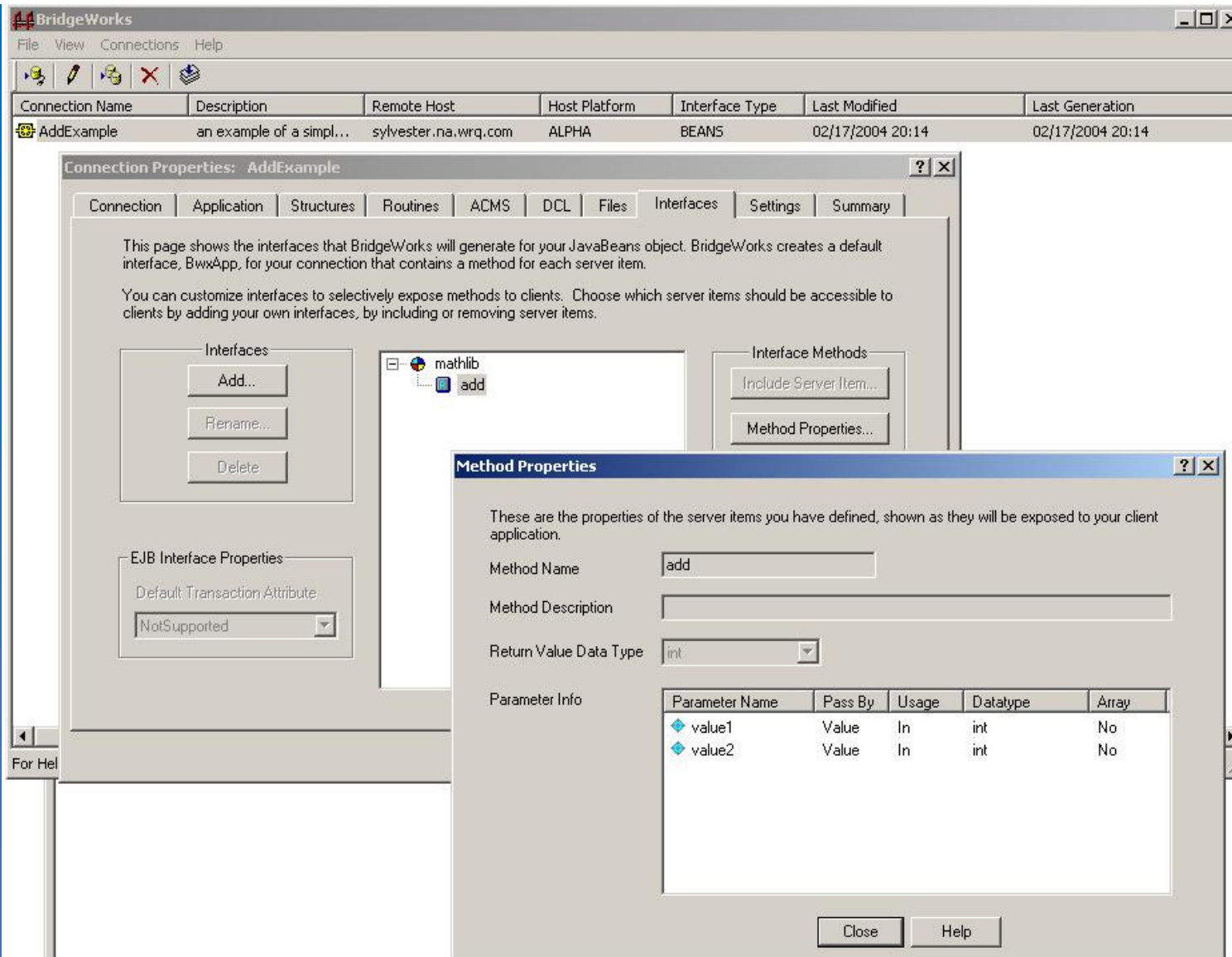
Freight Claims Composite Application Architecture Overview



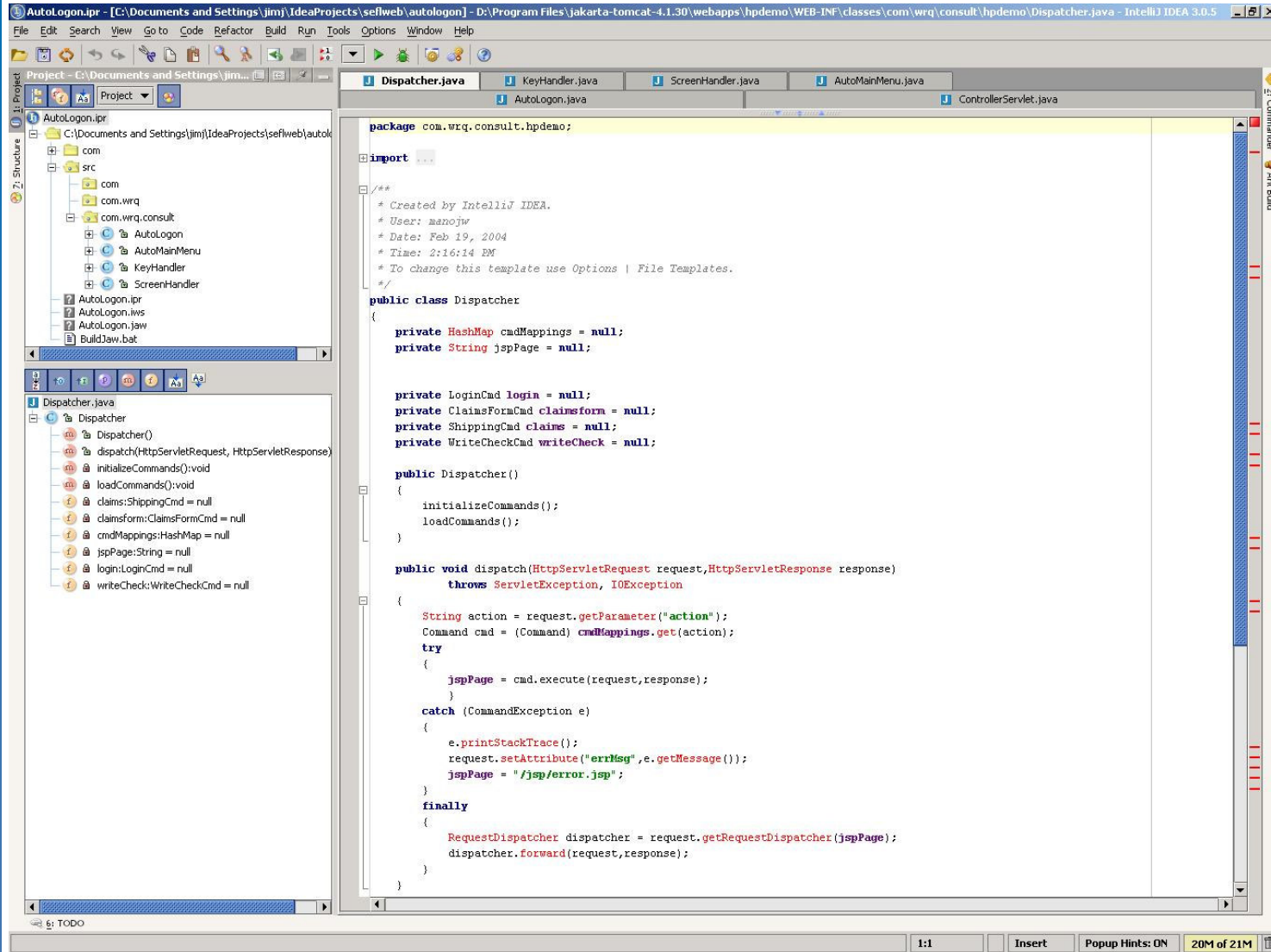
Building the DECforms Application Model



Building the Bridgeworks Component



Building the Java Components



Component Management WRQ Verastream Repository

The screenshot displays the WRQ Verastream Repository Manager application. The interface is divided into three main sections:

- Left Panel (Repositories):** A tree view showing the hierarchy of repositories. Three red circles highlight specific areas:
 - The top circle highlights the 'DecForms HP Verastream Demo' repository.
 - The middle circle highlights the 'AddExample.CmathlibBean' component under the 'Java Repository'.
 - The bottom circle highlights the 'SEFL DATA ON SYLVESTER' repository.
- Right Panel (Contents of Component 'AddExample.CmathlibBean'):** A table listing the components and their types. The table has two columns: 'Name' and 'Type'.
- Annotations:** Three red arrows point from text labels to the highlighted areas in the left panel:
 - 'DecForms Application Model Components' points to the top circle.
 - 'Java Components' points to the middle circle.
 - 'RMS Data Tables' points to the bottom circle.

Name	Type
add	Method
equals	Method
getClass	Method
hashCode	Method
new	Method
new_1	Method
new_2	Method
new_3	Method
notify	Method
notifyAll	Method
remove	Method
toString	Method
wait	Method
wait_1	Method
wait_2	Method
java.lang.Class	Object Reference
java.lang.Object	Object Reference

Assembling the Composite Application Verastream Process Integrator

The screenshot displays the WRQ Verastream Process Integrator application window. The title bar indicates the file path: `D:\Program Files\VeraStream\Programs\hpdemo\shipping.lgc`. The interface is divided into several panes:

- Scripts Components Functions Data Objects:** A tabbed menu at the top left.
- Find items:** A search bar within the components pane.
- Components and document structures:** A tree view on the left showing a hierarchy of components. The 'Shipping' component is expanded, showing sub-components like 'getClaimInfo', 'writeCheck', 'CLAIMS_NOT_FOUND_EXCEPTION', 'AccountInfo', 'Select', 'Update', 'connectException', 'inputColumnException', 'methodException', 'outputColumnException', 'performTableProcedureException', 'procedureException', 'tableException', 'terminalAttributesException', 'AddExample_CmathlibBean', 'add', 'equals', 'getClass', 'hashCode', 'new', 'new_1', 'new_2', 'new_3', 'notify', 'notifyAll', 'remove', 'toString', 'wait', 'wait_1', 'wait_2', 'BwxServerException', 'Exception', 'InterruptedException', 'RemoteException', 'Balance', 'Cash', 'CashWithdrawal', and 'Check'.
- Script: claimByProNumber.get:** The main workspace showing the script logic. It starts with 'No inputs for this script'. The logic includes:
 - A check: `length(strip(Claims_inq_ref_num)) > 0`.
 - A SQL query: `claims_acctg . SELECT COLUMNS: ALL CRITERIA: claims_acctg_pro_number == strip(Claims_inq_ref_num) ORDER: claims_acctg_pro_number`.
 - A loop structure with a 'claims_acctg . FIRST' and 'claims_acctg . NOCURR' block, followed by a 'claims_acctg . NEXT' block and a 'claimRecord . add' block.
 - An 'ON FAILURE' section with a 'Raise error condition' block.
 - Another 'ON FAILURE' section with an exception block: `EXCEPTION="CLAIMS_NOT_FOUND_EXCEPTION" EXCEPTIONSTRING="Invalid Pro Number entered"`.
 - Ends with 'No outputs for this script'.
- Properties Description Cross Reference:** A tabbed menu at the bottom left.
- Edit Description:** A text area at the bottom right showing the script's creation details: 'created by: MANOJW-VW2K\manojw' and 'date: Dec 06 2003, 04:49:16'.



Ready, Set, Go!!
Let's Get Started!!

Here's what we will do....

OpenVMS Authentication Component

Southwestern Freight Lines - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Forward Stop Search Favorites History Print

Address <http://localhost:8080/hpdemo/jsp/login.jsp> Go

Links [Best of the Web](#) [Channel Guide](#) [Free Hotmail](#) [Internet Explorer News](#) [Internet Start](#) [RealPlayer](#) [Customize Links](#) [Windows Media](#) [Windows](#) [Computer](#) [WRQ](#)

SFL
SOUTHEASTERN FREIGHT LINES

CLAIM PROCESSING SYSTEM LOGIN

Login ID

Password

Login Clear

Single sign on is available....

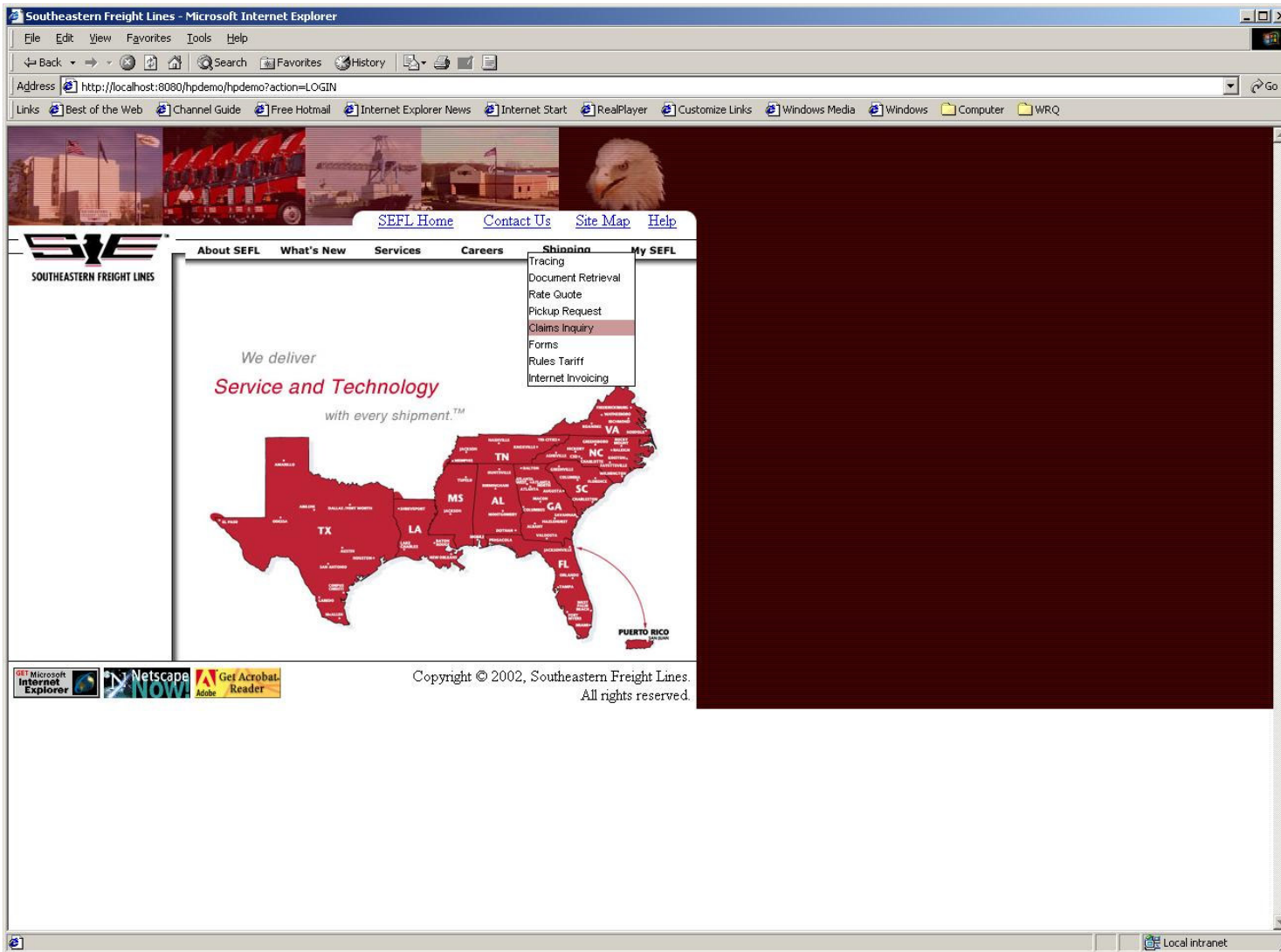
POWERED BY
WRQ verastream

No matter what platform the web application is running on,
you are authenticated against the cluster wide sysuaf.dat

Enter your OpenVMS username and password!

Done Local intranet

Initiating the Claims Inquiry Process



Query for Damaged Freight Claim Data—OpenVMS RMS Data Files

Southeastern Freight Lines | Claims Inquiry - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Forward Stop Home Search Favorites History Print Mail

Address <http://localhost:8080/hpdemo/hpdemo?action=GETCLAIMSFORM> Go

Links Best of the Web Channel Guide Free Hotmail Internet Explorer News Internet Start RealPlayer Customize Links Windows Media Windows Computer WRQ

SEFL
SOUTHEASTERN FREIGHT LINES

Shipping
Tracing
Document Retrieval
Rate Quote
Pickup Request
Claims Inquiry
Forms
Rules Tariff
Internet Invoicing

SEFL Corporate Office:
420 Davega Road
Lexington, SC 29073
800.637.7335
803.794.8131 Fax

Contact Us Site Map Help

About SEFL What's New Services Careers Shipping My SEFL

Claims Inquiry

Pro Number ☐ Reference Number:
SEFL Claim Number ☒

Please select either pro number or claim number and then enter the appropriate number.
Note: Do not enter dashes on claim numbers or pro numbers.

Copyright © 2002, Southeastern Freight Lines.
All rights reserved.

Microsoft Internet Explorer Netscape Now! Get Acrobat Reader

Done Local intranet

Paying the Damage Claim

Southeastern Freight Lines | Claims Inquiry - Microsoft Internet Explorer

Address: http://localhost:8080/hpdemo/hpdemo?action=GETCLAIMS

Links: Best of the Web, Channel Guide, Free Hotmail, Internet Explorer News, Internet Start, RealPlayer, Customize Links, Windows Media, Windows, Computer, WRQ

SEFL Home Contact Us Site Map Help

SEFL
SOUTHEASTERN FREIGHT LINES

Shipping
Tracing
Document Retrieval
Rate Quote
Pickup Request
Claims Inquiry
Forms
Rules Tariff
Internet Invoicing

SEFL Corporate Office:
420 Davaega Road
Lexington, SC 29073
800.637.7335
803.794.8131 Fax

Claims Inquiry

Pro Number:	601036531	Claim Amount:	\$ 92.80
Claim Number:	0300004	Date Received:	01/02/2003
Claimant Number:	172-71397	Status:	PAID
Name:	UNISOURCE		
Shipper Name:	UNISOURCE		
Consignee Name:	UNISOURCE		
Total Paid:	\$ 92.80	Amount Declined:	
Date Paid:	01/03/2003	Date Declined:	
SEFL Check Number:	079441		
Decline Reason:			

Write Check

Copyright © 2002, Southeastern Freight Lines. All rights reserved.

Microsoft Internet Explorer Netscape Get Acrobat Reader

Done Local intranet

Call DECForms
Application via
Java/JSP

Verastream Host Adapter Session Pools

The screenshot displays the WRQ Verastream Session Monitor application. The main window is titled "WRQ Verastream Session Monitor" and includes a menu bar with "Configure", "Help", and "Exit". On the left, there is a sidebar with a dropdown menu set to "localhost", buttons for "Disconnect AADS" and "Disconnect Server", a "View Active Sessions" button, and a "DECforms" dropdown with a "View Session Pool" button. The main area is titled "Idle Sessions in DECforms" and features a "Refresh" button. Below this is a table with the following data:

Previous Session Id	Time of Last Use	Connected to Host	Current Entity
6	Apr 09 13:47:16 2004	☒	PersonalCheckingMenu
4		☒	
3		☒	PersonalCheckingMenu
2		☒	
1		☒	

Below the main window is a smaller window titled "Session 6". It has buttons for "Play", "Record", and "Close". The session content area shows a terminal-like interface with the following text:

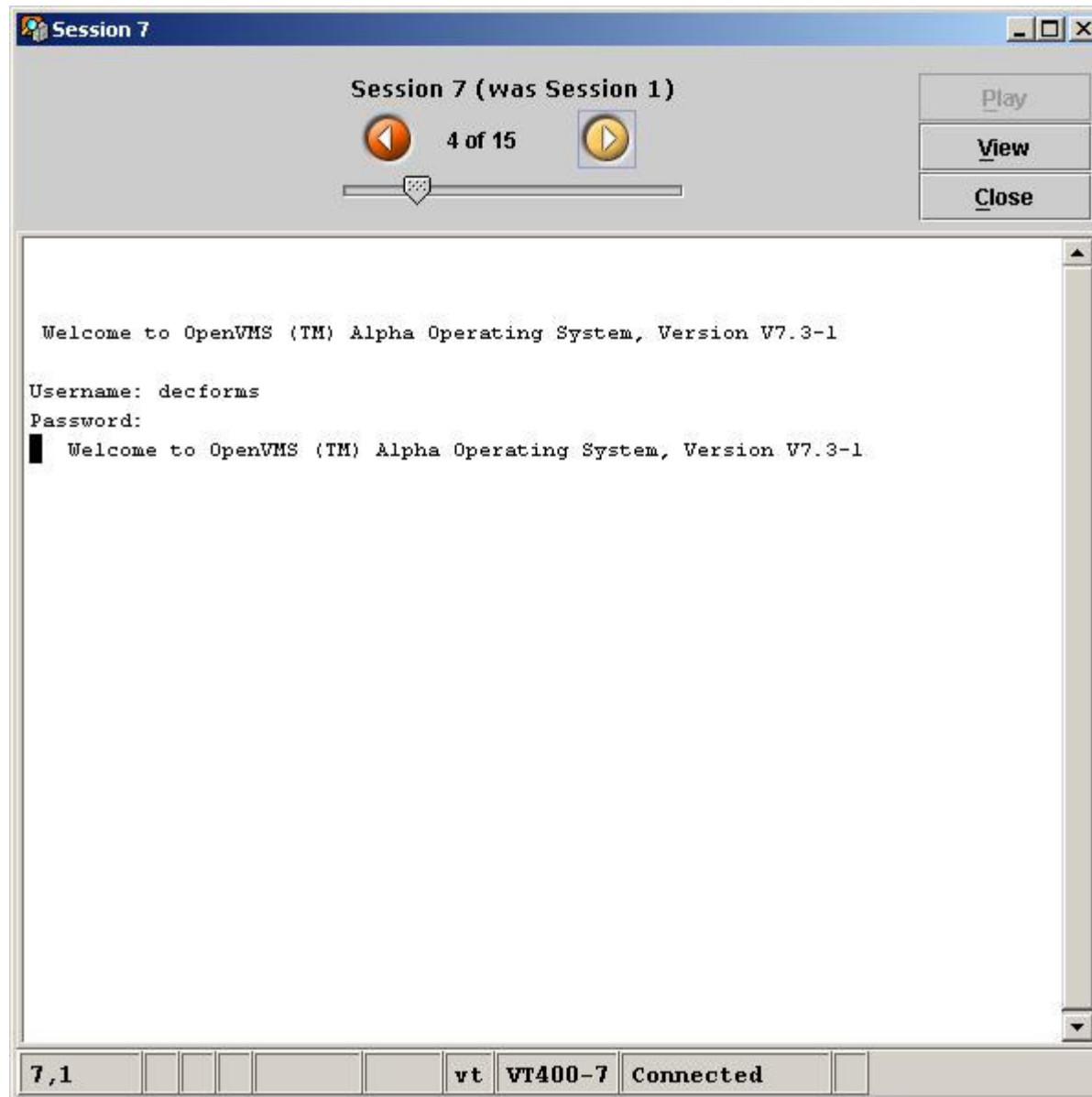
```
Personal Checking Account
Personal Checking Account
Write a check... Deposit... Cash... Transfer... Review... Exit

Balances: Checking:    $39.14
           Savings:    $212.55

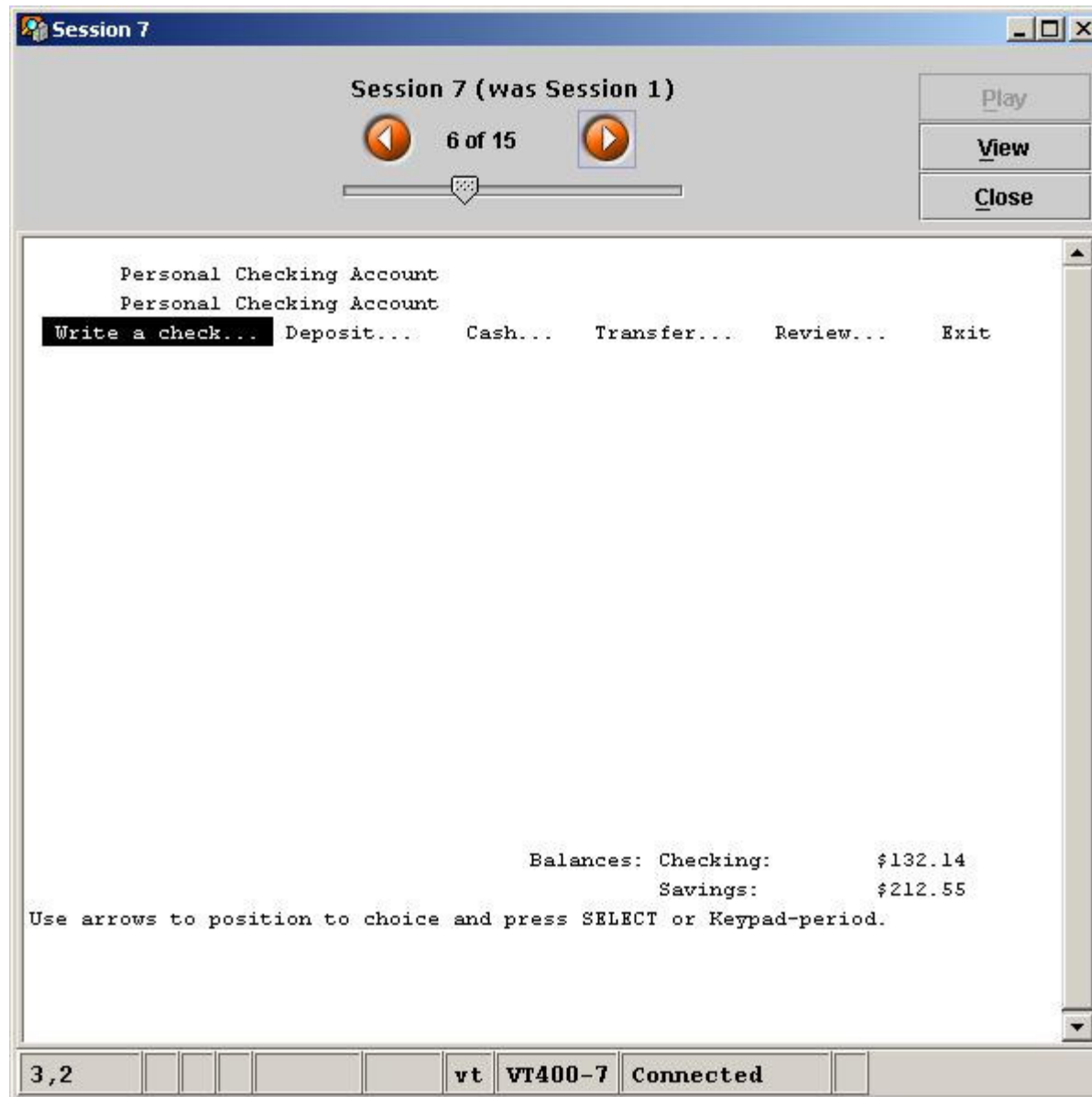
Press F10 or PF1-E to leave the application.
Check Paid: *****93.00
```

At the bottom of the application, a status bar shows "3,2" on the left and "vt VT400-7 Connected" on the right.

Behind the Scenes— DECforms Checking Transaction Execution via Verastream Host Adapter Component



Behind the Scenes— DECforms Checking Transaction Execution via Verastream Host Adapter Component



Behind the Scenes— DECforms Checking Transaction Execution via Verastream Host Adapter Component

Session 7

Session 7 (was Session 1)

9 of 15

Play View Close

Personal Checking Account
Personal Checking Account

Write a Check

Judith M. Jones 14
1425 Gould Avenue
Birmingham, VT 99999
Home (800)555-1999 Office (800)555-1111

Date: April 9, 2004

Pay to [REDACTED] Amount \$*****0.00

Memo _____

FIRST NATIONAL BANK Account 00532

Fill in the fields and terminate with F10 or PF1-E. Cancel with F8 or PF1-Q.

Balances: Checking: \$132.14
Savings: \$212.55

Use arrows to position to choice and press SELECT or Keypad-period.
Press F10 or PF1-E to leave the application.

12,11 vt VT400-7 Connected

Behind the Scenes— DECforms Checking Transaction Execution via Verastream Host Adapter Component

Session 7

Session 7 (was Session 1)

14 of 15

Play View Close

Personal Checking Account
Personal Checking Account

Write a Check

Judith M. Jones 14
1425 Gould Avenue
Birmingham, VT 99999
Home (800)555-1999 Office (800)555-1111

Date: April 9, 2004

Pay to UNISOURCE Amount \$*****93.00

Memo Claims Refund

FIRST NATIONAL BANK Account 00532

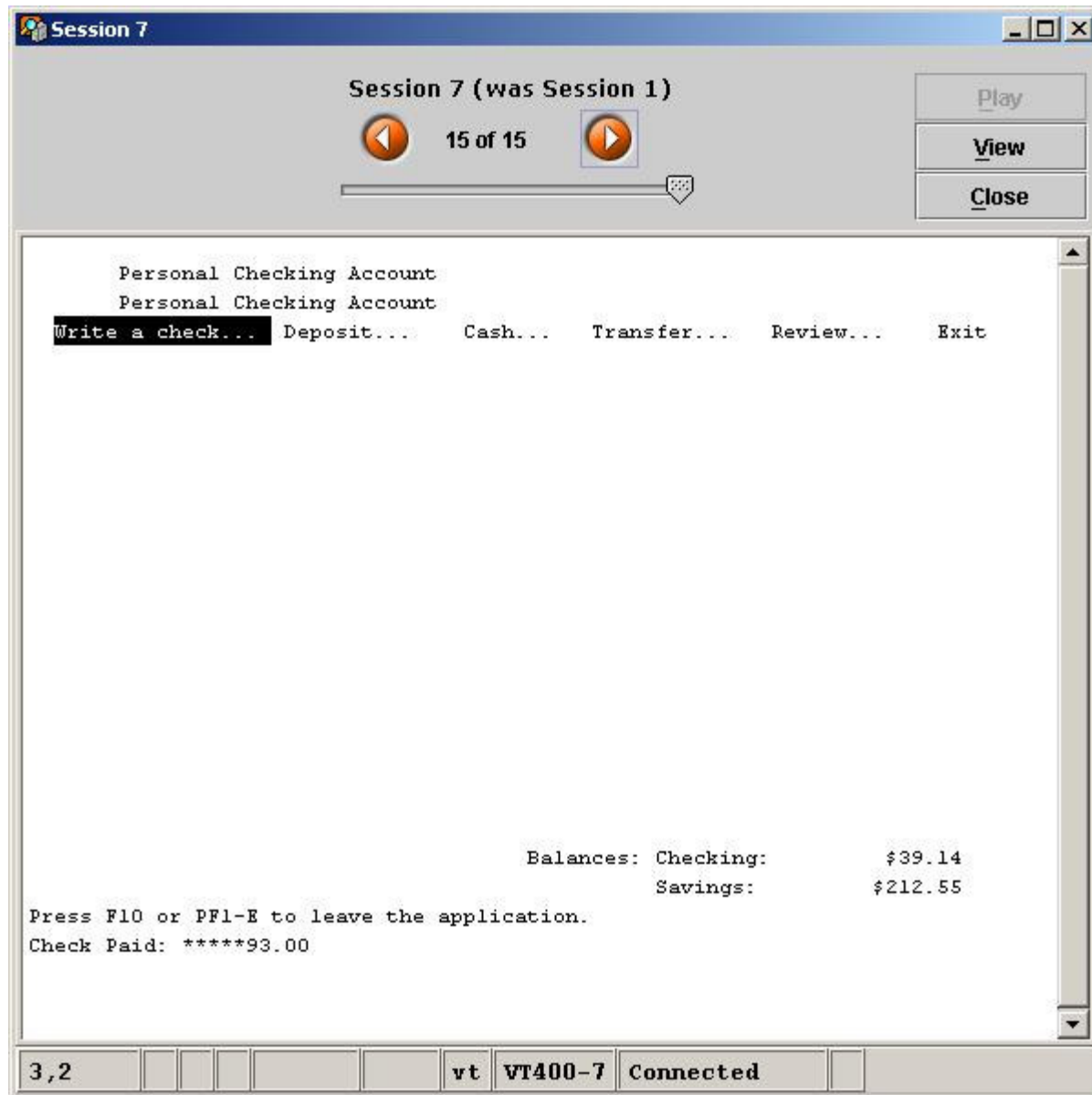
Fill in the fields and terminate with F10 or PF1-E. Cancel with F8 or PF1-Q.

Balances: Checking: \$132.14
Savings: \$212.55

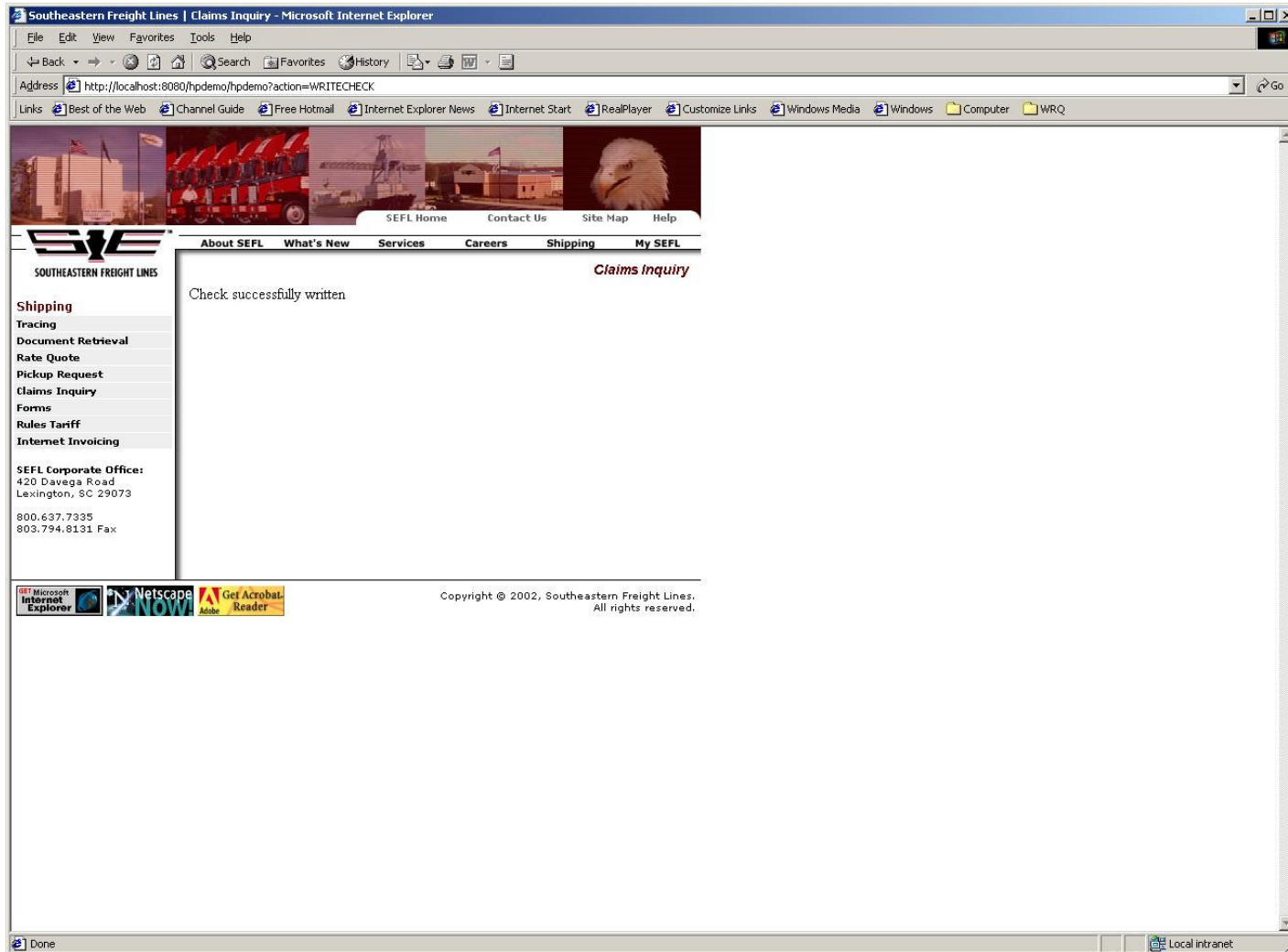
Use arrows to position to choice and press SELECT or Keypad-period.
Press F10 or PF1-E to leave the application.

14,44 vt VT400-7 Connected

Behind the Scenes— DECforms Checking Transaction Execution via Verastream Host Adapter Component

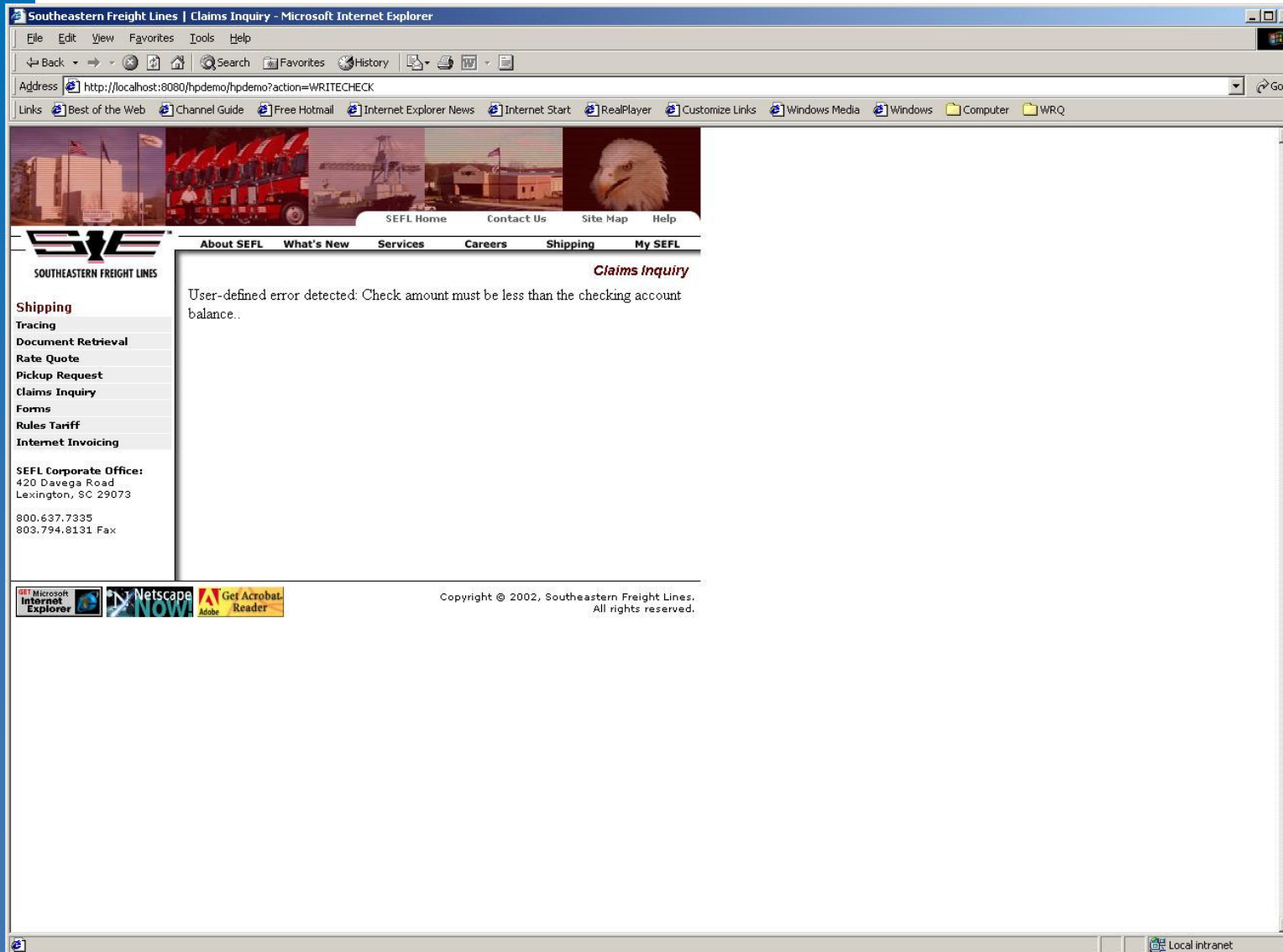


A Successful Transaction!



Ooops!!

Robust Error/Exception Handling



Composite Application Advantages

Faster, better results!

- Designed to work the way your business does...Allows you to tailor applications and integration to YOUR business needs instead of tailoring your business to the software!
- Reuse the thousands of business functions/ applications already proven and in production
- Minimize testing and risk...No alterations to existing code
- Highly flexible... Stands alone or complements other EAI infrastructures!
- Highly scalable... Built-in load balancing
- Proven success with many customers



Freight Claims Payment Composite Application

Hands-On Tutorial Step by Step

Verastream Composite Application Overview

Freight Claims Payment—Composite Application

Combines/reuses/integrates:

- RMS Data used/maintained by existing OpenVMS Cobol Applications
- FMS/Fortran Character Based Checking Account Application wrapped with Verastream Host Integrator
- A legacy OpenVMS application written in C and wrapped with HP Bridgeworks to create a Java callable interface
- A modern easy-to-use WEB user interface developed using Java/JSP

Verastream Development Process

Freight Claims Payment

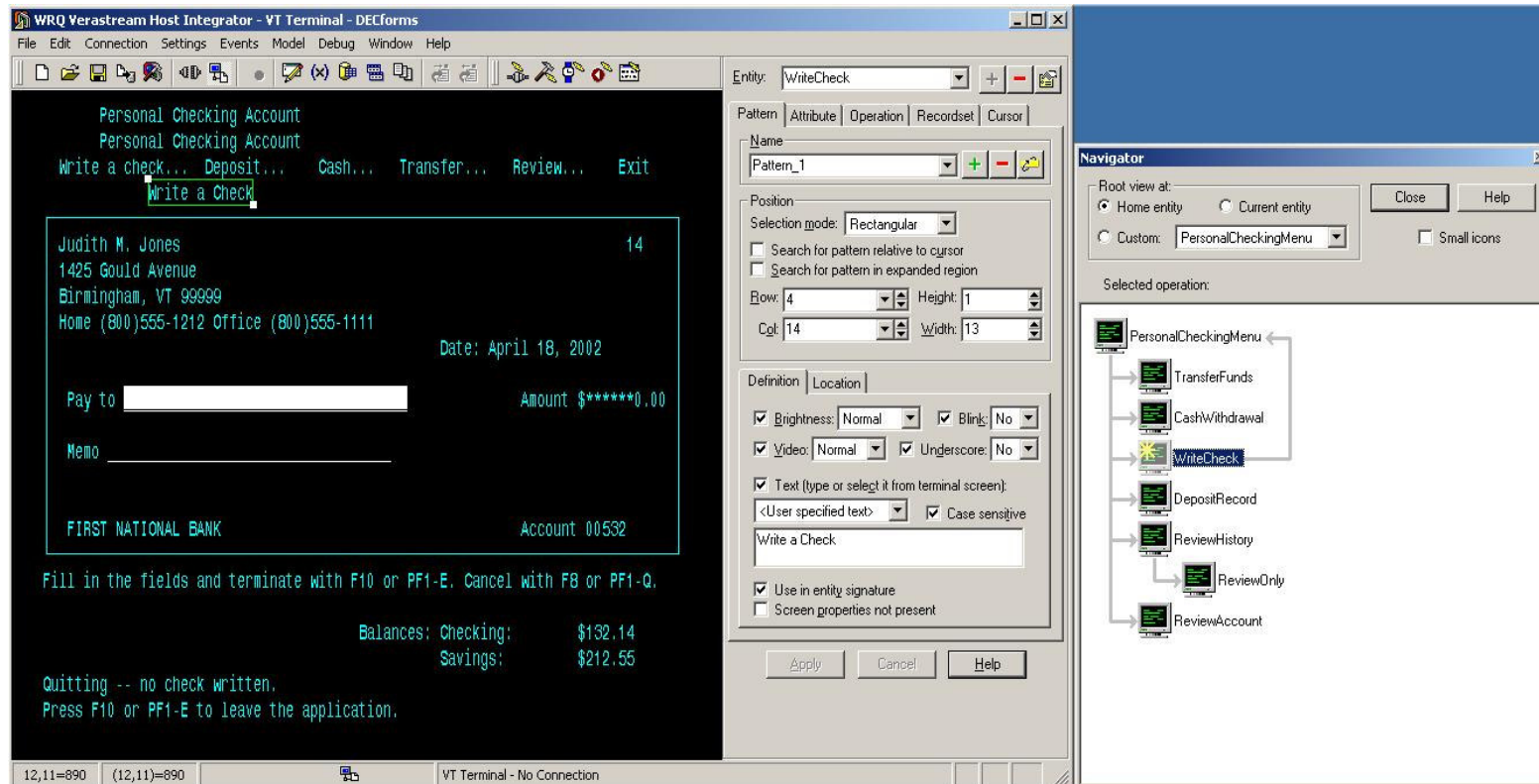
Composite Application Development Process

Five simple steps:

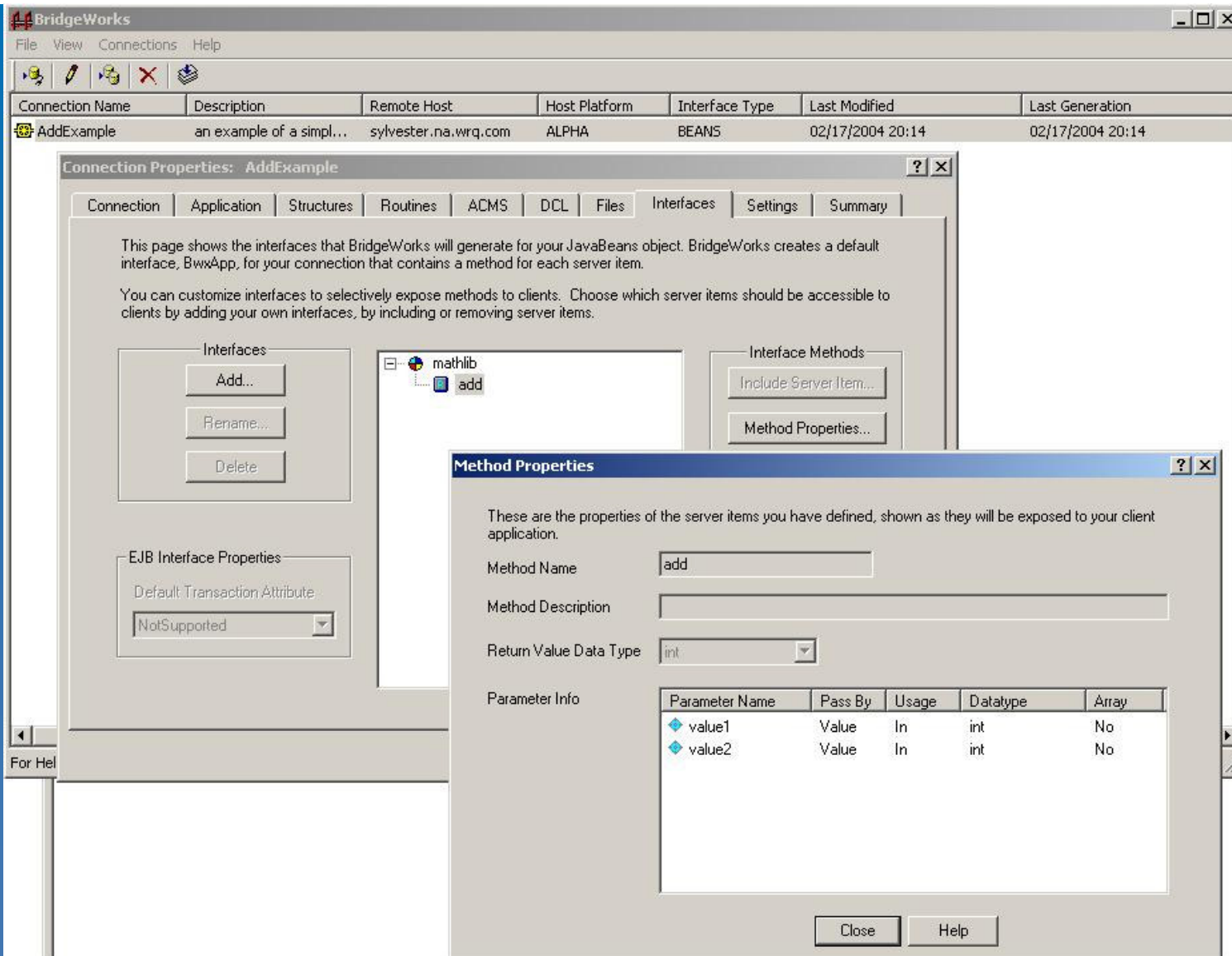
- Build the components (DECForms Verastream Host Integrator model, Bridgeworks Java Bean and Verastream Shipping Component)*
- Add the components to the Repository
- Assemble the composite application
- Deploy the composite application
- Test the composite application
- Mission Accomplished!

* - To save time, for this tutorial an example DECForms Checking model, Bridgeworks Java Bean and Java Servlets/JSP's have been pre-built for you)

Build the components Verastream Host Integrator Model



Build the components Bridgeworks Java Bean



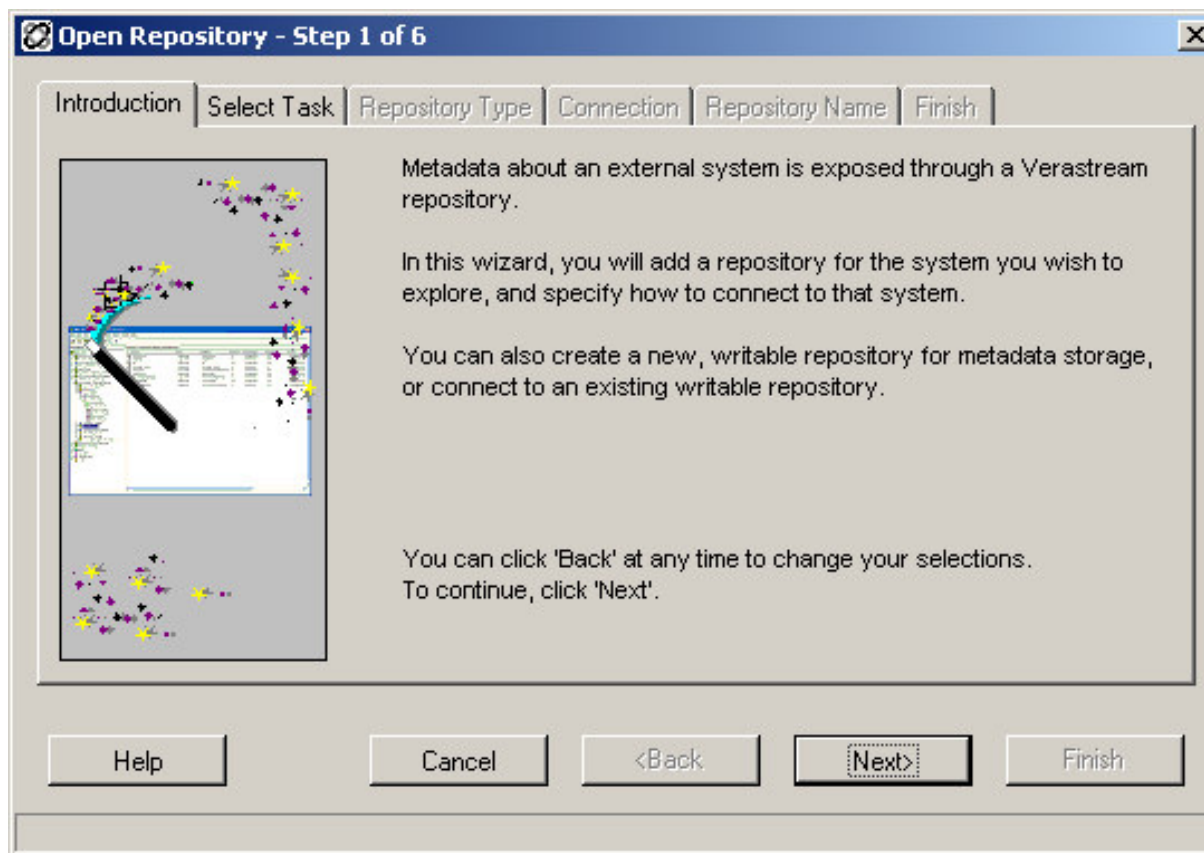
Build the components Verastream Shipping Component – Composite Application

- ☐ Open a Table Repository (To import metadata of RMS data files)
- ☐ Open a Verastream Host Integrator Model Repository
- ☐ Open a Java Repository
- ☐ Create a Verastream Repository
- ☐ Define the Shipping Component
- ☐ Create “claimsdd” module file
- ☐ Realize (Implement) the Component
- ☐ Register (deploy) the Component
- ☐ Test the Component

Building the Shipping Component

Open a Table Repository

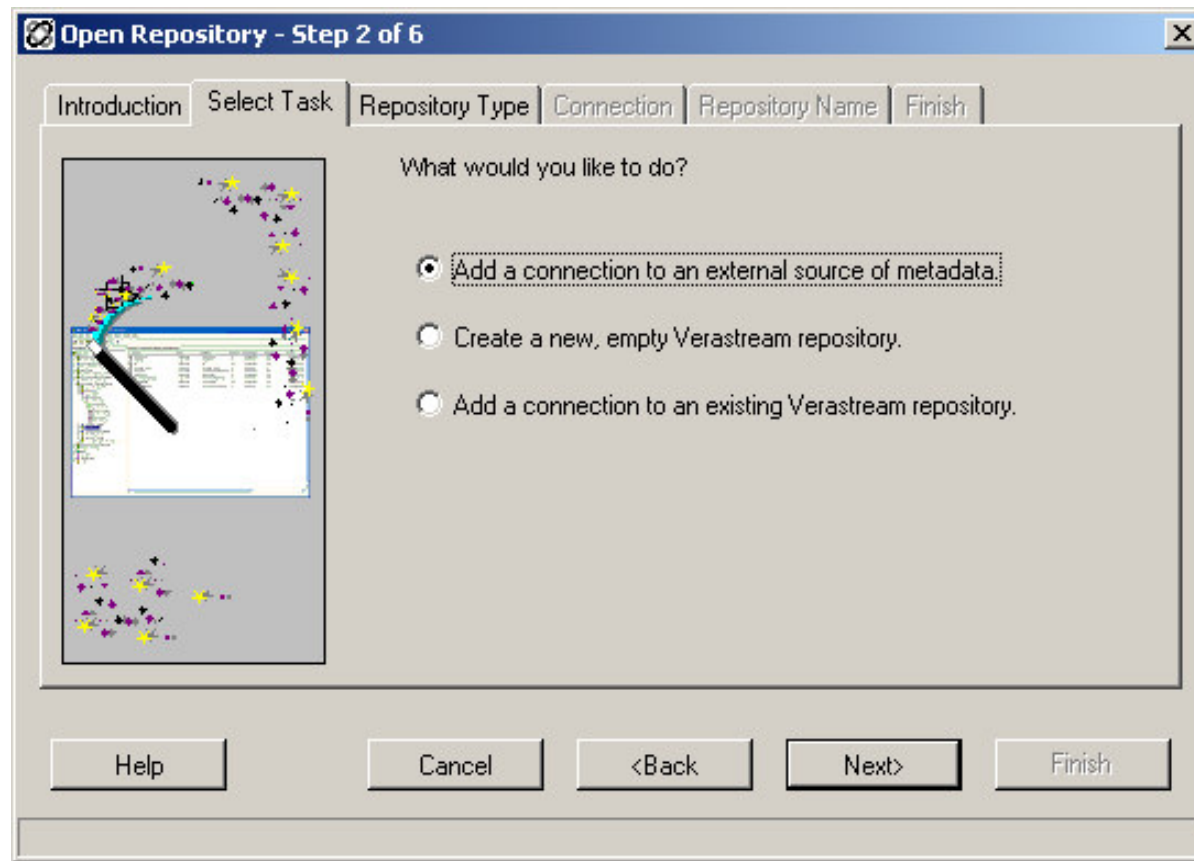
- Launch Verastream Repository Manager
- Select the menu File → Open Repository



Building the Shipping Component

Open a Table Repository

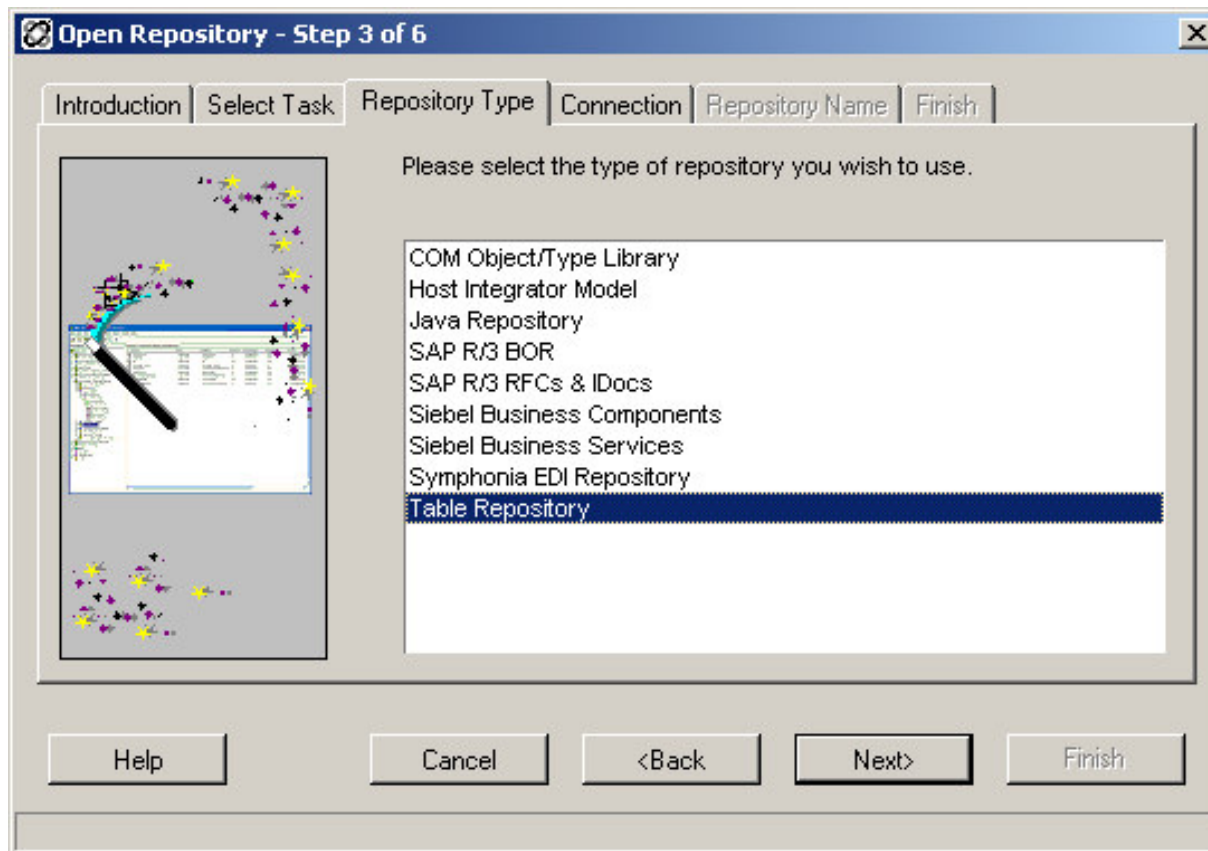
- Add a connection to an external source of metadata



Building the Shipping Component

Open a Table Repository

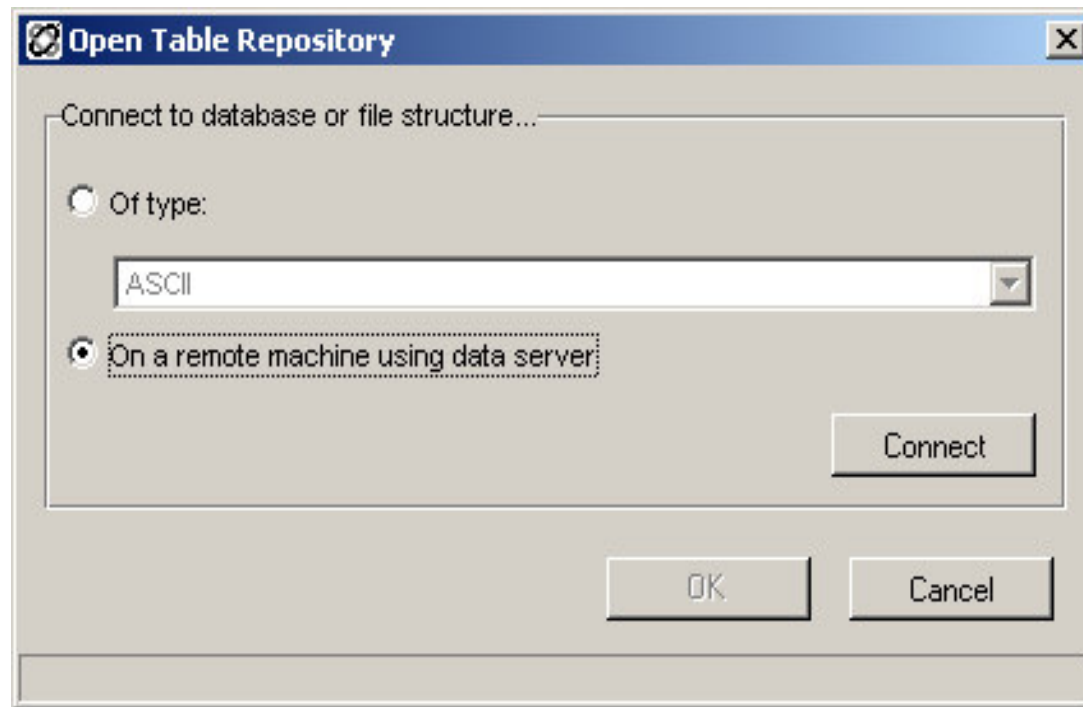
- Select the Table Repository



Building the Shipping Component

Open a Table Repository

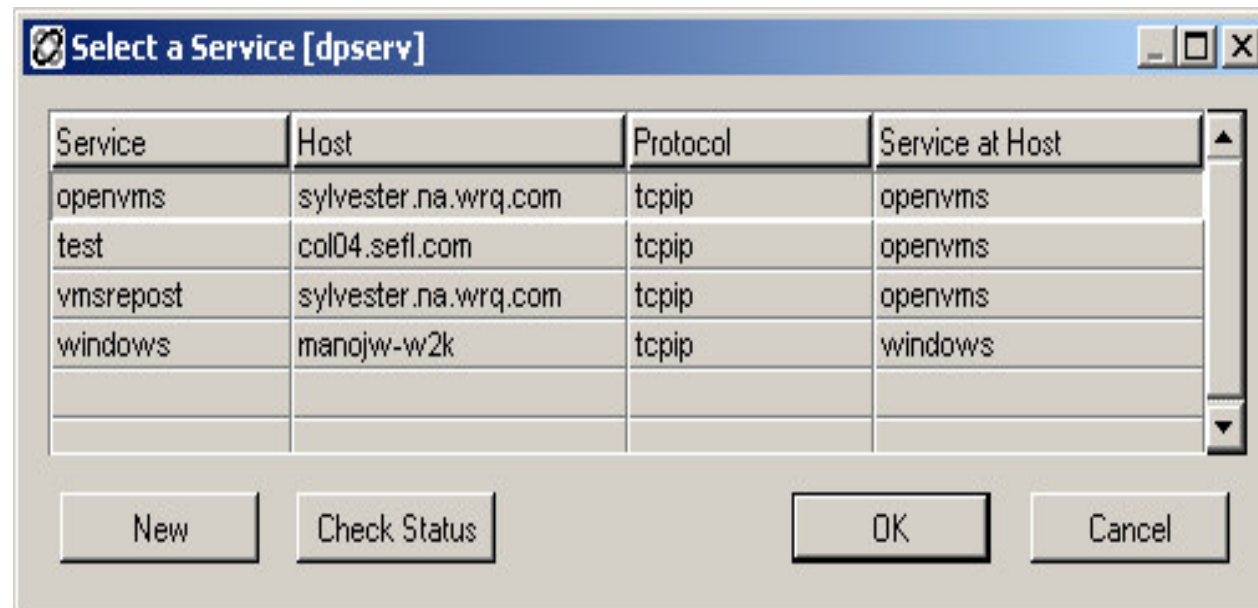
- Connect to the database using data server



Building the Shipping Component

Open a Table Repository

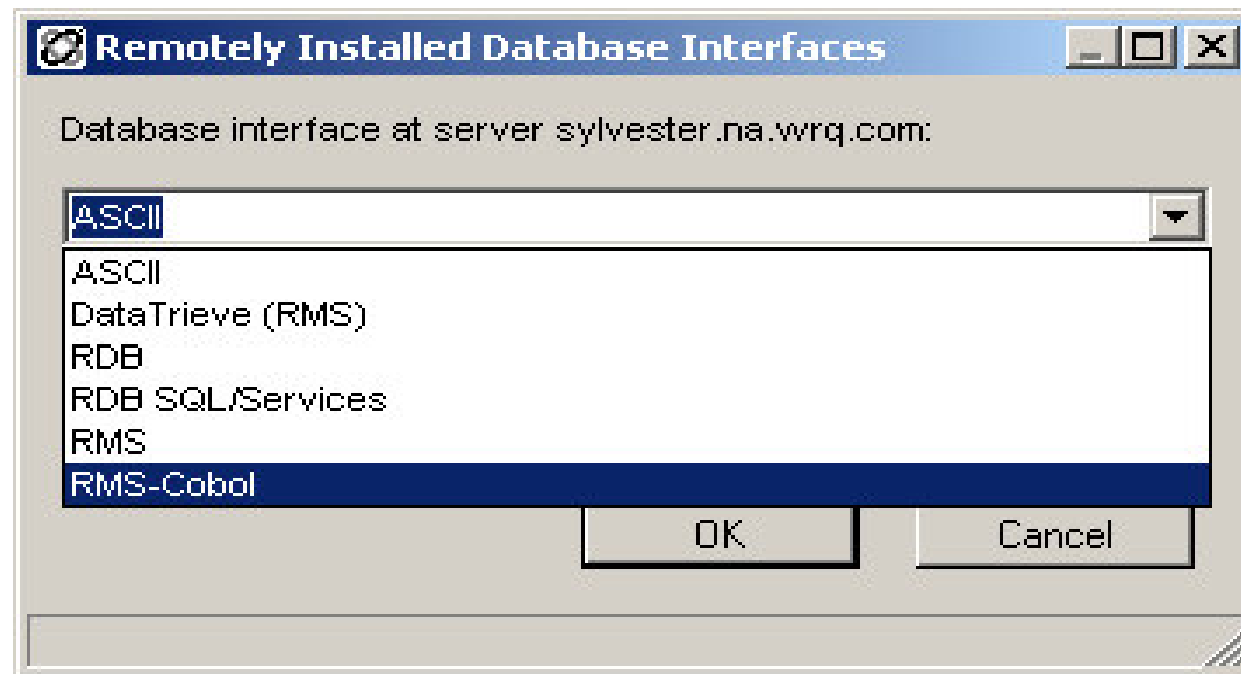
- Select the client service



Building the Shipping Component

Open a Table Repository

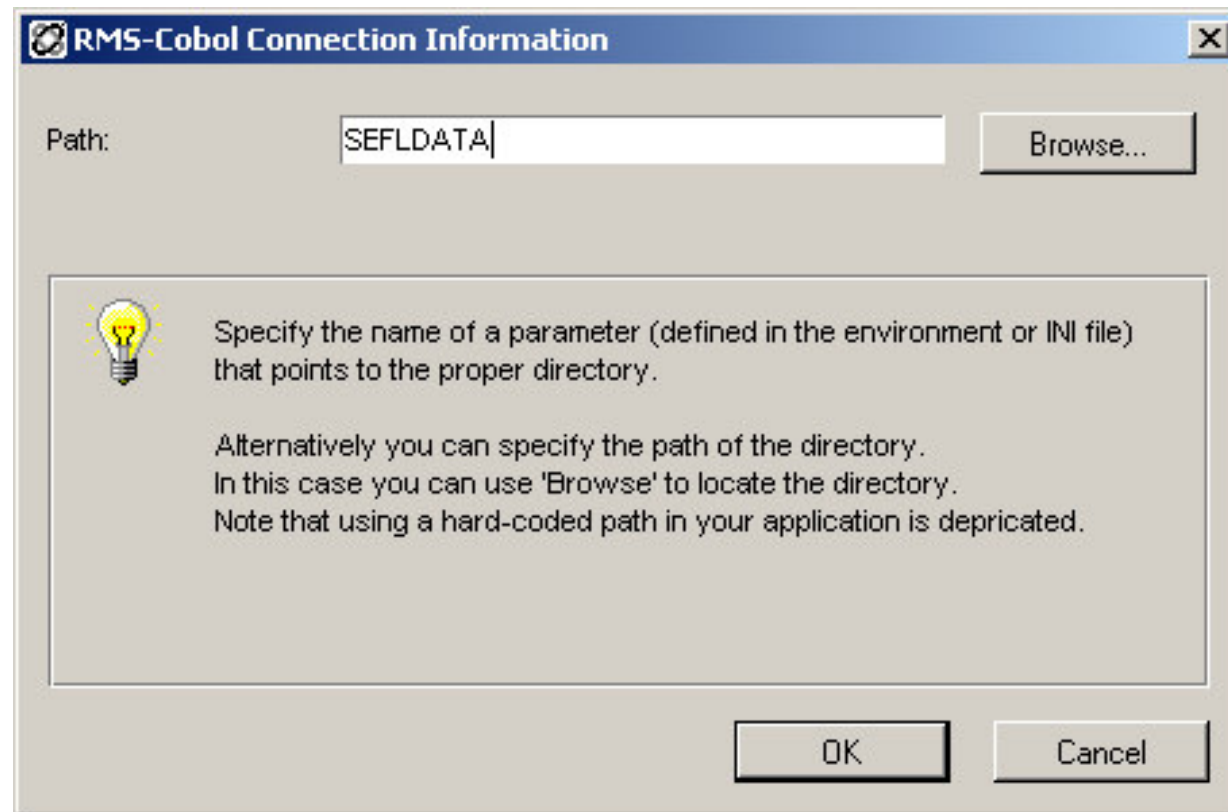
- Select the database interface type - (RMS-Cobol)



Building the Shipping Component

Open a Table Repository

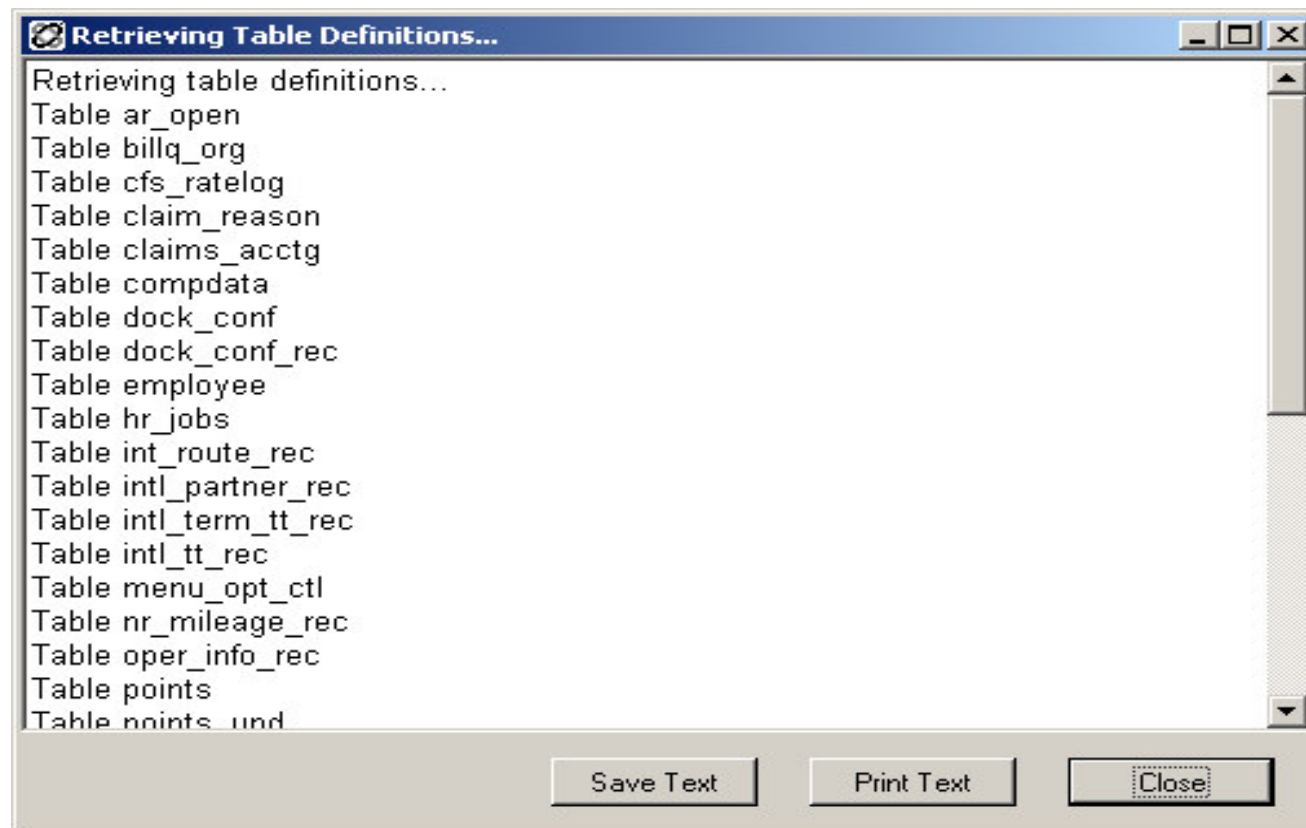
- Specify path to the Cobol copy book



Building the Shipping Component

Open a Table Repository

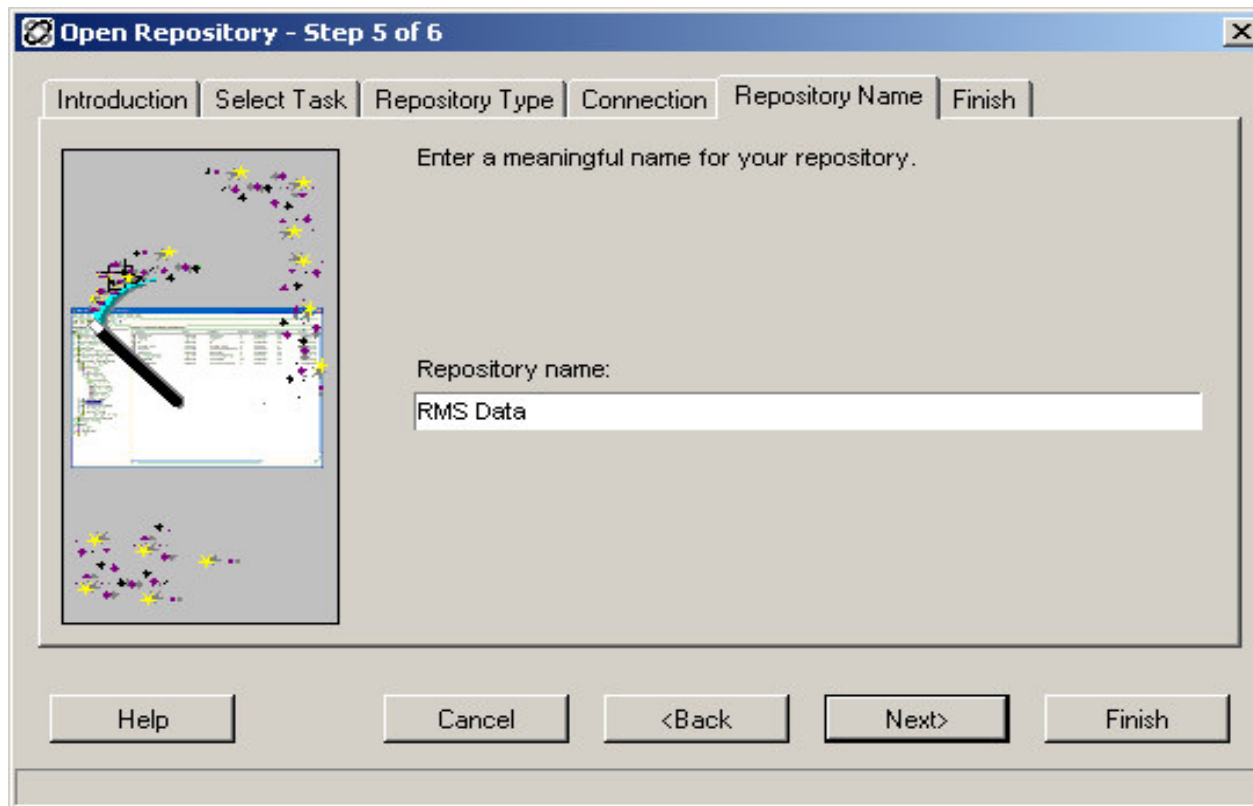
- View the list of tables whose metadata is being imported



Building the Shipping Component

Open a Table Repository

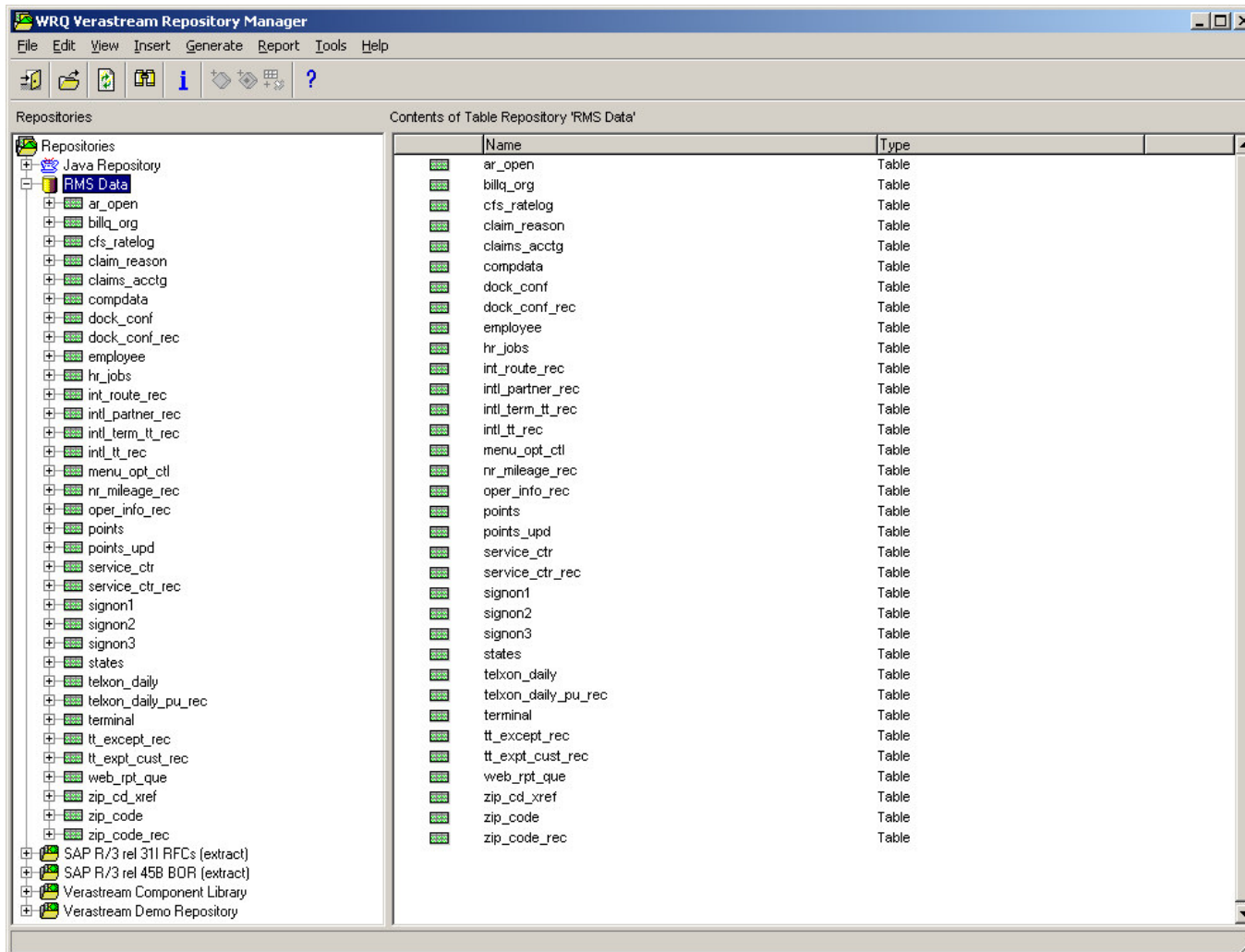
- Name the Repository
- Click Finish



Building the Shipping Component

Open a Table Repository

- View the table repository



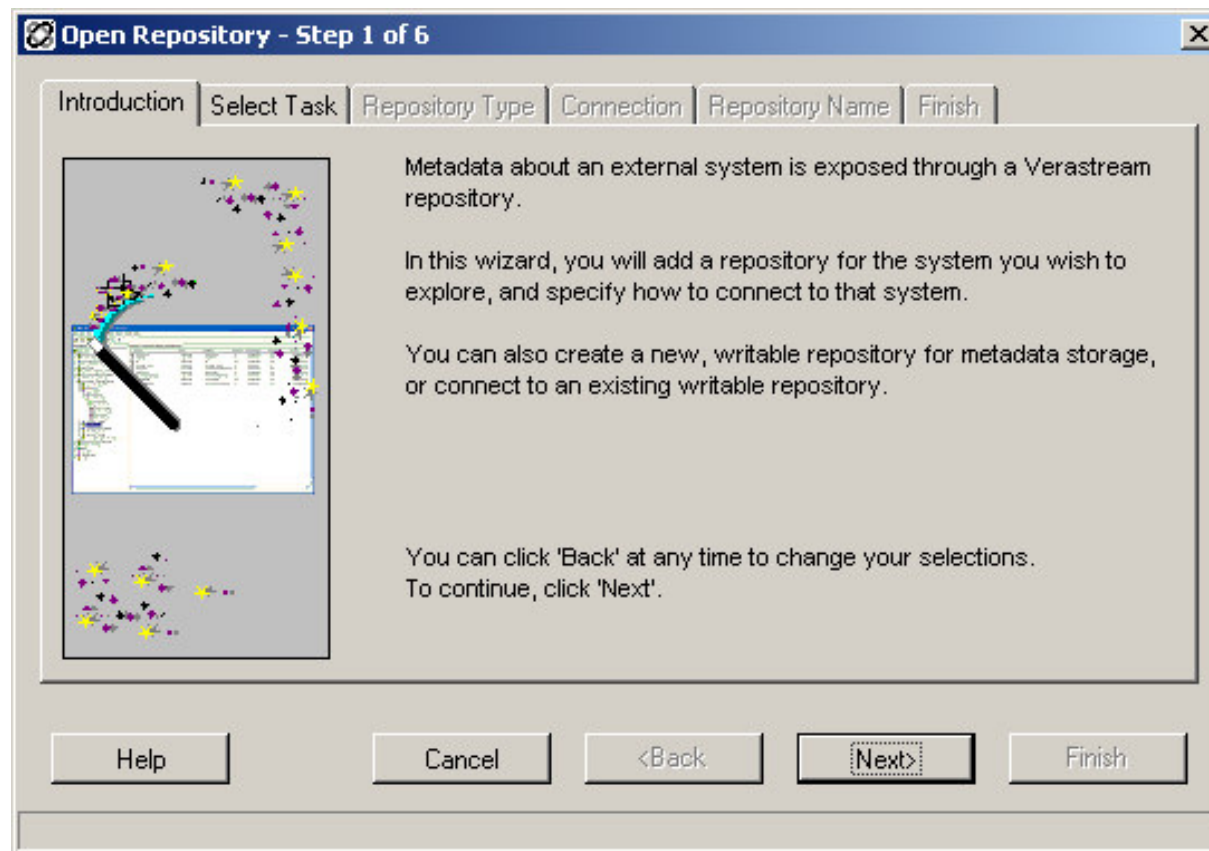
Build the components Verastream Shipping Component – Composite Application

- ☒ Open a Table Repository (To import metadata of RMS data files)
- ☐ Open a Verastream Host Integrator Model Repository
- ☐ Open a Java Repository
- ☐ Create a Verastream Repository
- ☐ Define the Shipping Component
- ☐ Create “claimsdd” module file
- ☐ Realize (Implement) the Component
- ☐ Register (deploy) the Component
- ☐ Test the Component

Building the Shipping Component

Open a Host Integrator Model Repository

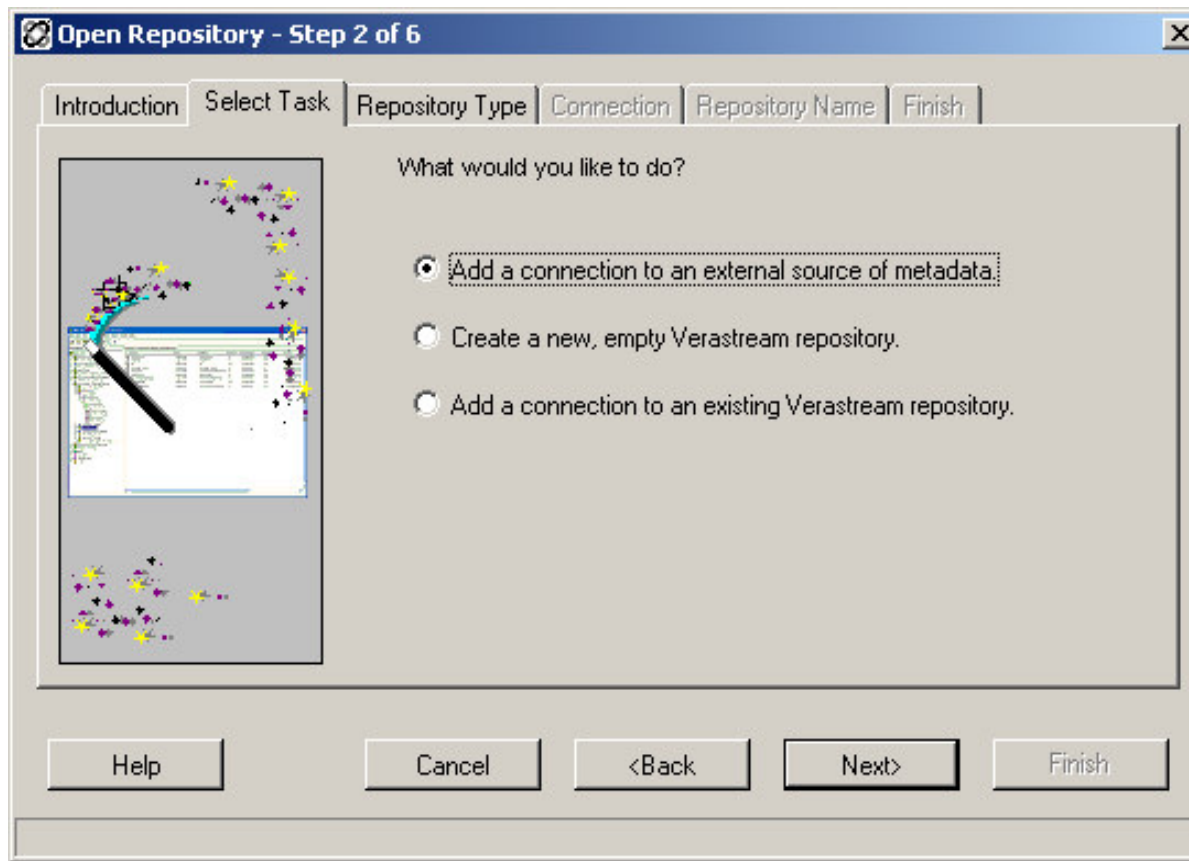
- Go to Verastream Repository Manager
- Select the menu File → Open Repository



Building the Shipping Component

Open a Table Repository

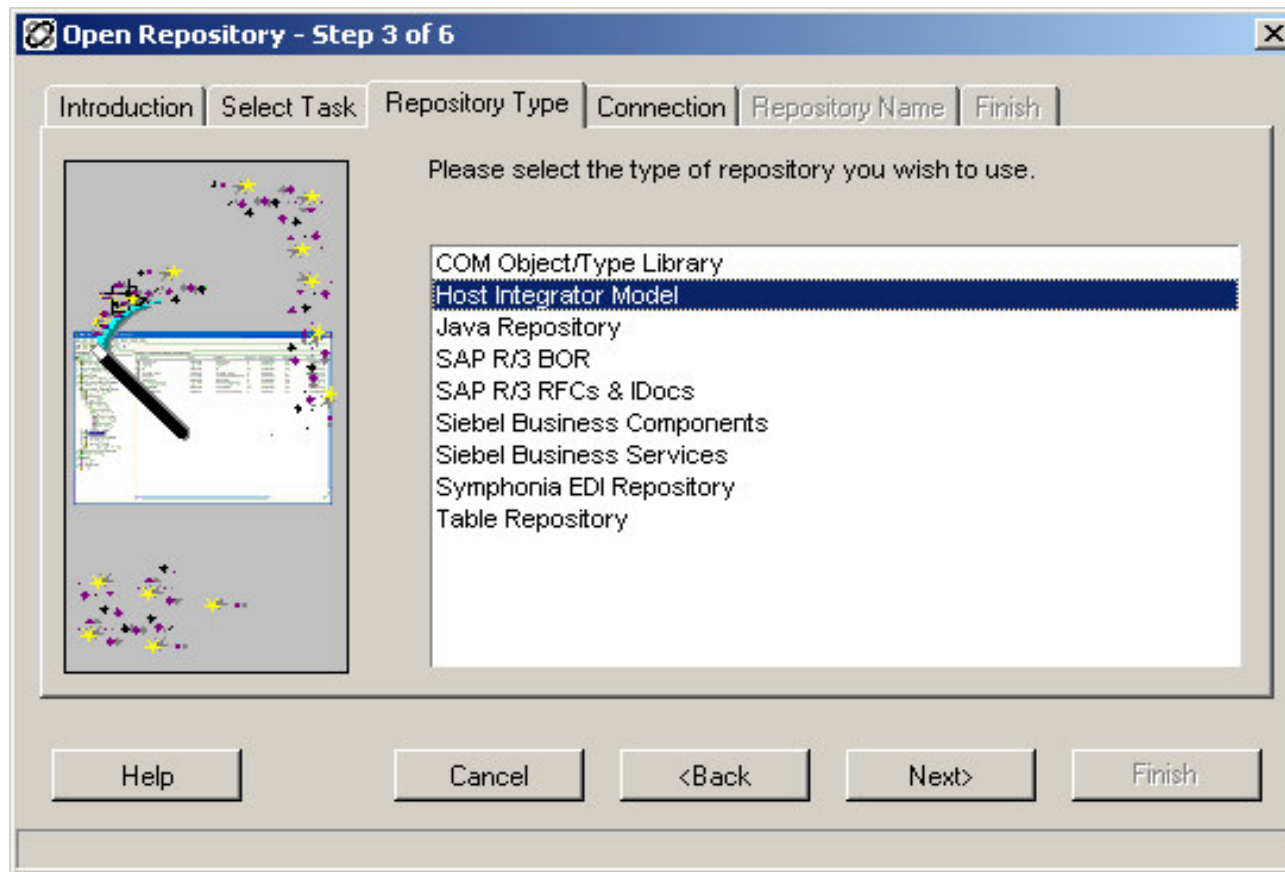
- Add a connection to an external source of metadata



Building the Shipping Component

Open a Table Repository

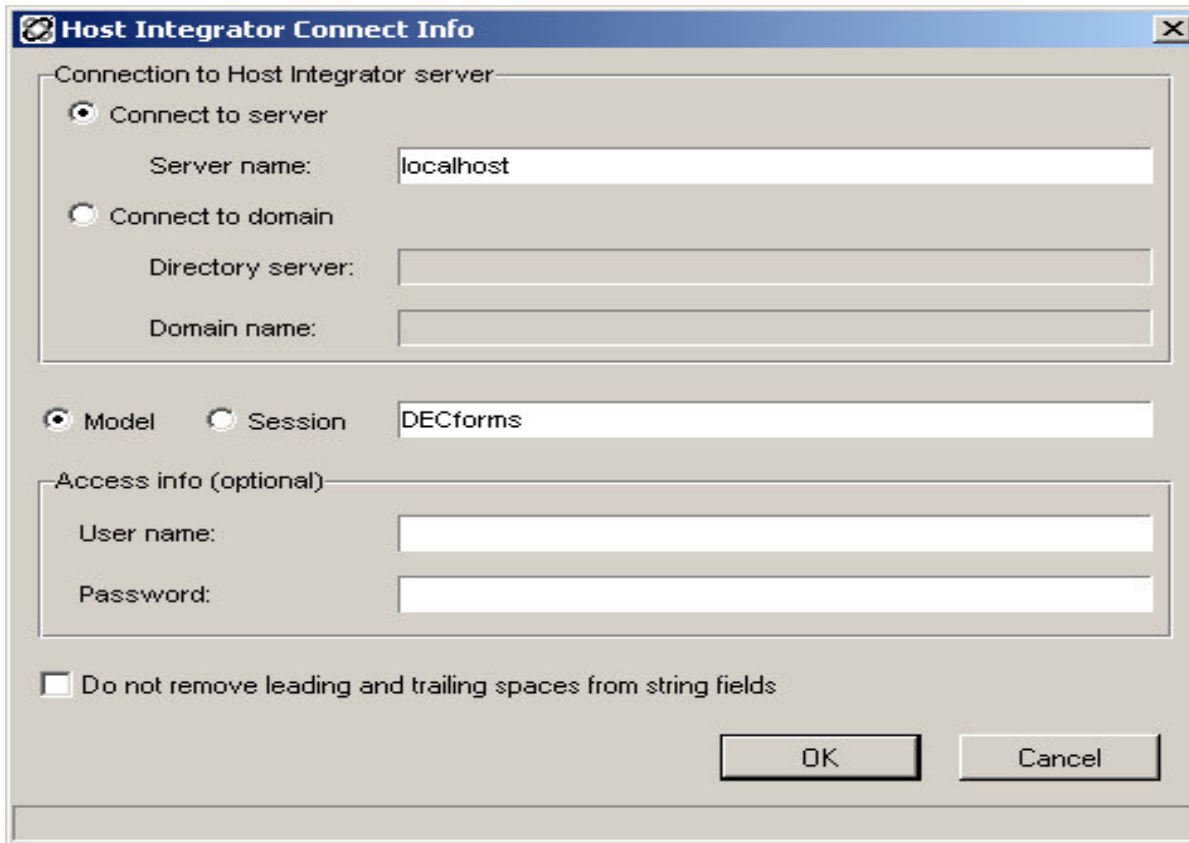
- Select Host Integrator Model



Building the Shipping Component

Open a Table Repository

- Provide the connection information to import the model metadata



The image shows a Windows-style dialog box titled "Host Integrator Connect Info". It contains several sections for configuring a connection. The first section, "Connection to Host Integrator server", has two radio buttons: "Connect to server" (which is selected) and "Connect to domain". Below "Connect to server" is a text field for "Server name" containing the text "localhost". Below "Connect to domain" are two text fields: "Directory server" and "Domain name". The second section has two radio buttons: "Model" (selected) and "Session". To the right of these is a text field containing "DECforms". The third section, "Access info (optional)", contains two text fields: "User name" and "Password". At the bottom, there is a checkbox labeled "Do not remove leading and trailing spaces from string fields" which is currently unchecked. At the very bottom are "OK" and "Cancel" buttons.

Host Integrator Connect Info

Connection to Host Integrator server

☒ Connect to server

Server name: localhost

☐ Connect to domain

Directory server:

Domain name:

☒ Model ☐ Session DECforms

Access info (optional)

User name:

Password:

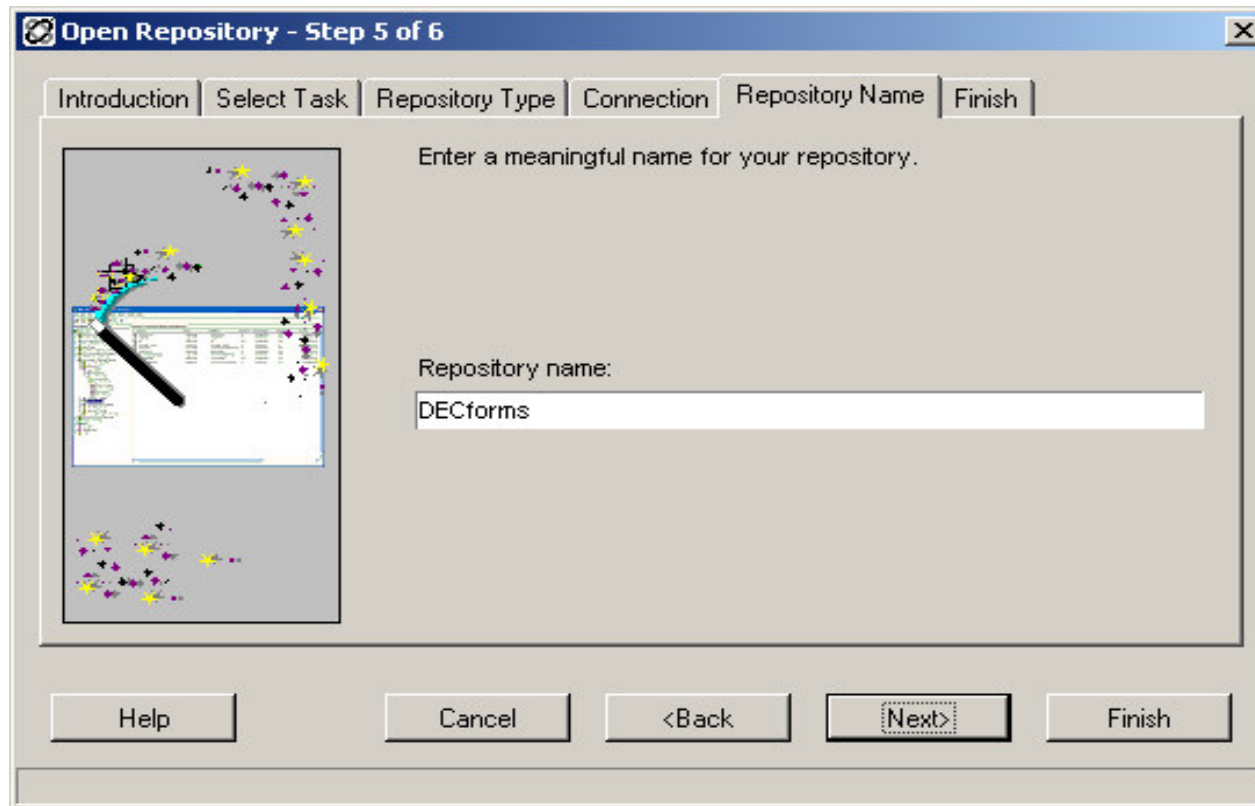
☐ Do not remove leading and trailing spaces from string fields

OK Cancel

Building the Shipping Component

Open a Table Repository

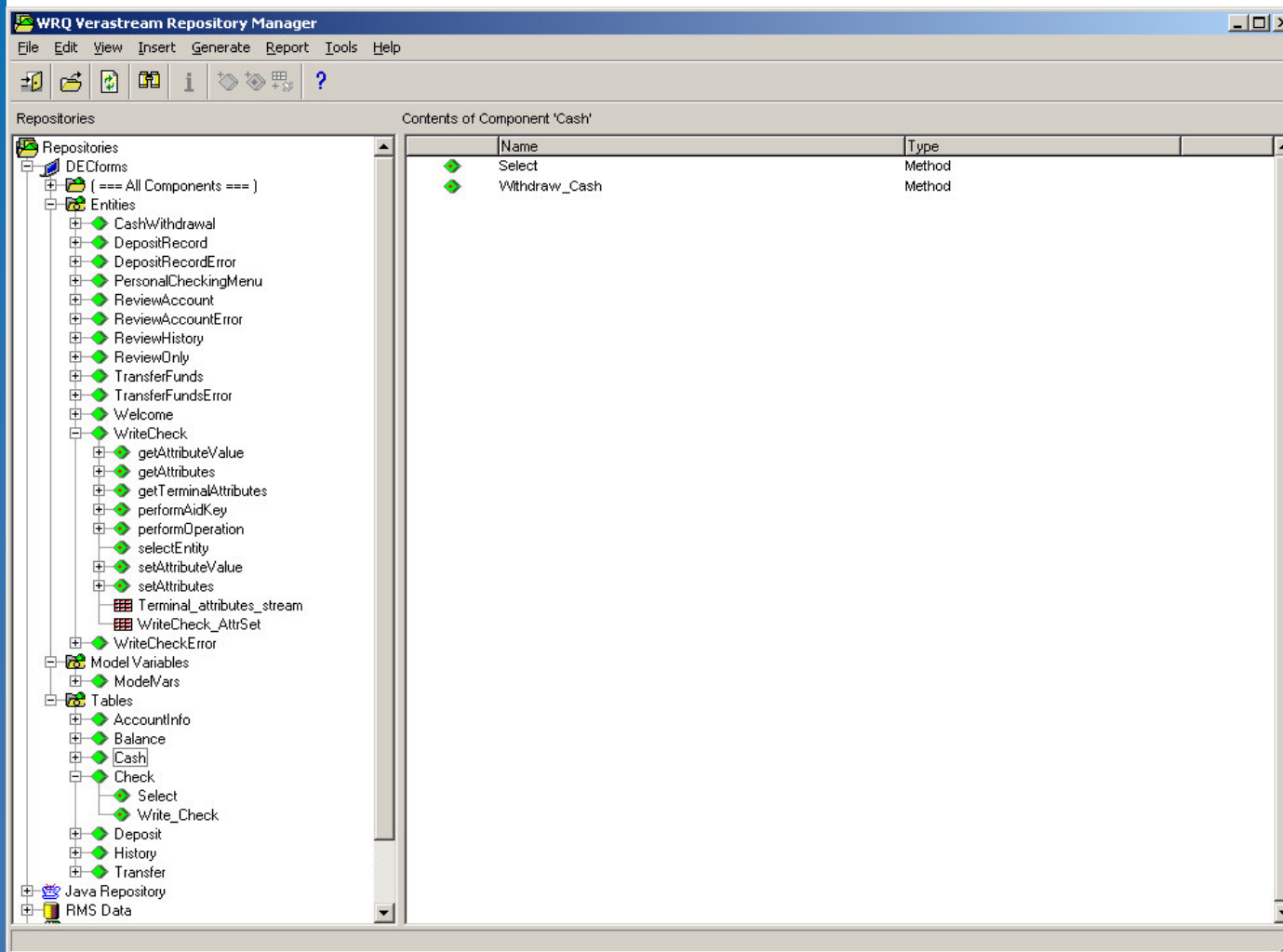
- Name the Repository
- Click Finish



Building the Shipping Component

Open a Table Repository

- View the Host Integrator Model Repository

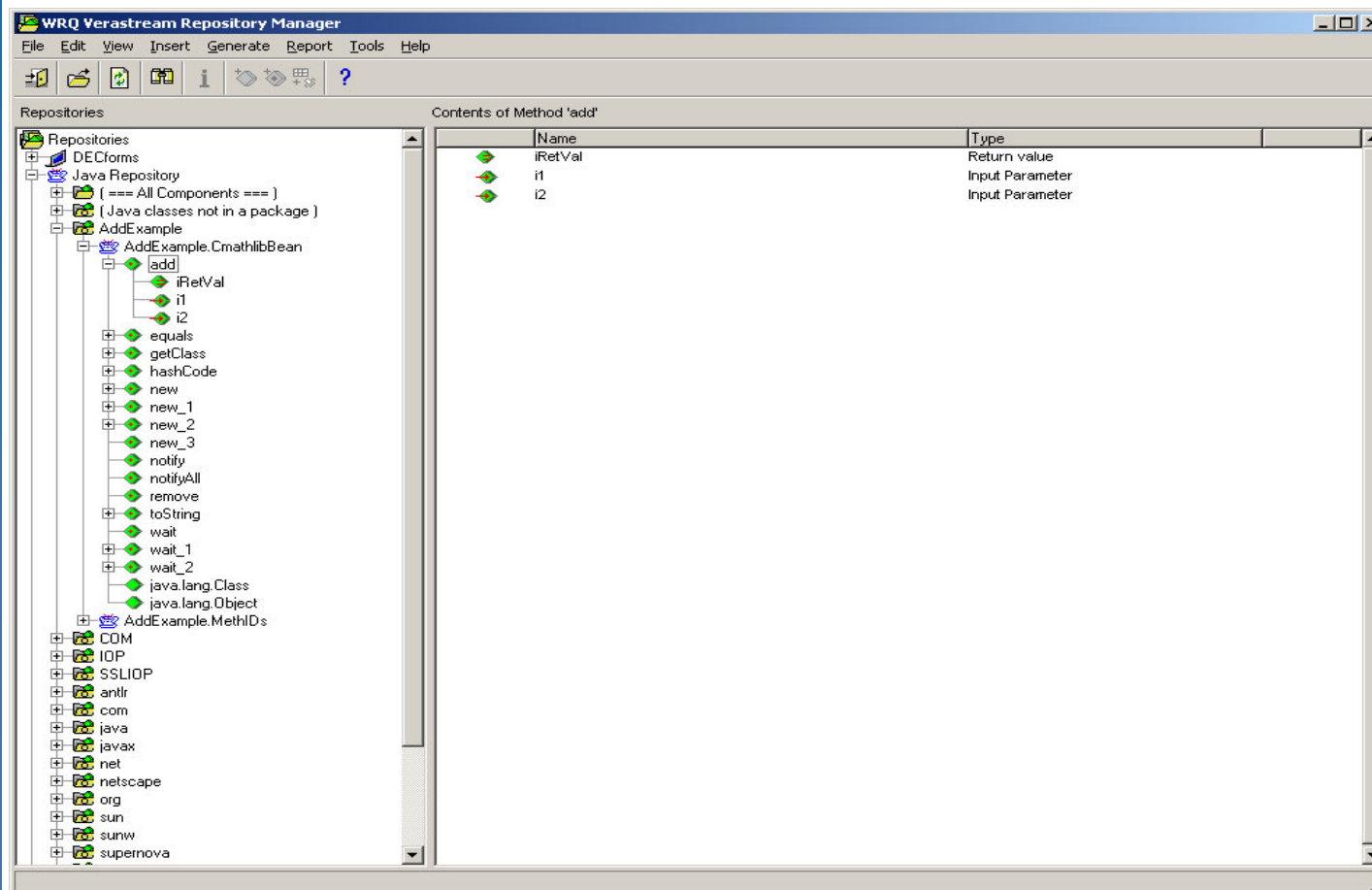


Build the components Verastream Shipping Component – Composite Application

- ☒ Open a Table Repository (To import metadata of RMS data files)
- ☒ Open a Verastream Host Integrator Model Repository
- ☐ Open a Java Repository
- ☐ Create a Verastream Repository
- ☐ Define the Shipping Component
- ☐ Create “claimsdd” module file
- ☐ Realize (Implement) the Component
- ☐ Register (deploy) the Component
- ☐ Test the Component

Building the Shipping Component Open a Java Repository

- Go to Verastream Repository Manager
- Expand the node Java Repository



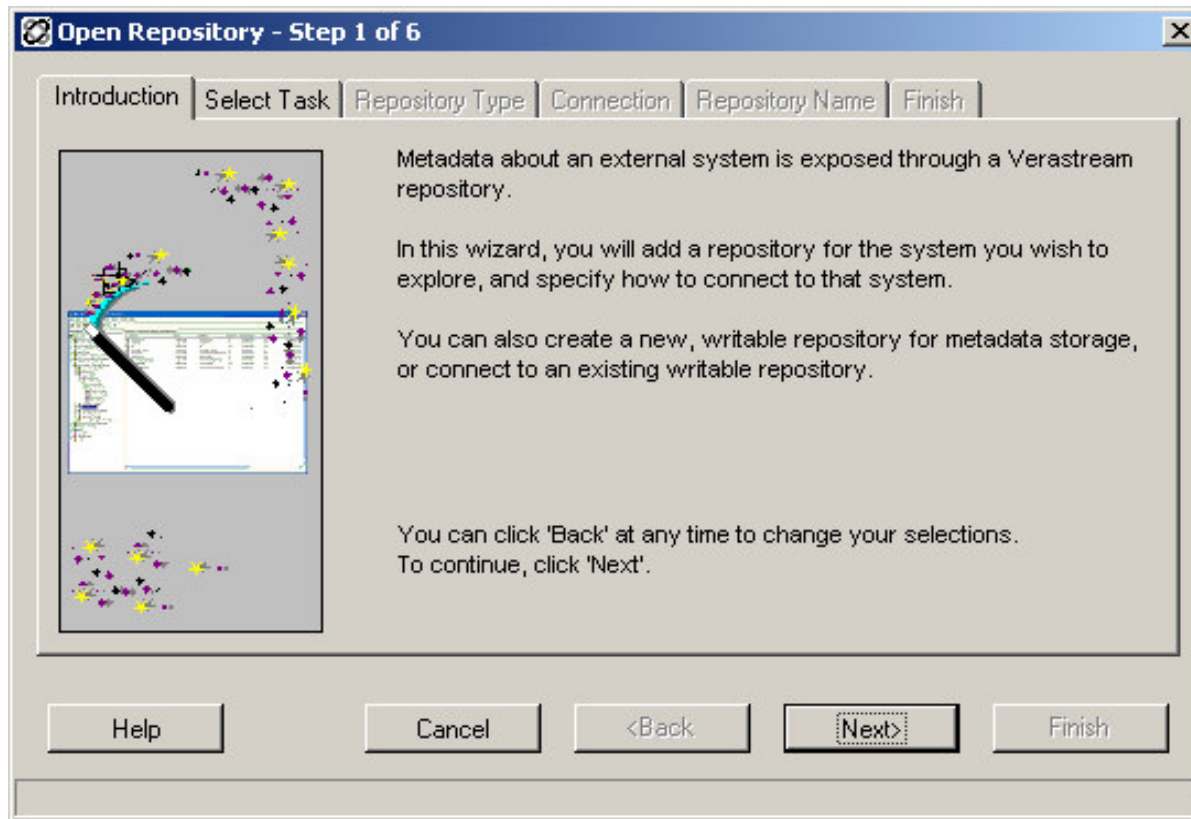
Build the components Verastream Shipping Component – Composite Application

- ☒ Open a Table Repository (To import metadata of RMS data files)
- ☒ Open a Verastream Host Integrator Model Repository
- ☒ Open a Java Repository
- ☐ Create a Verastream Repository
- ☐ Define the Shipping Component
- ☐ Create “claimsdd” module file
- ☐ Realize (Implement) the Component
- ☐ Register (deploy) the Component
- ☐ Test the Component

Building the Shipping Component

Create a Verastream Repository

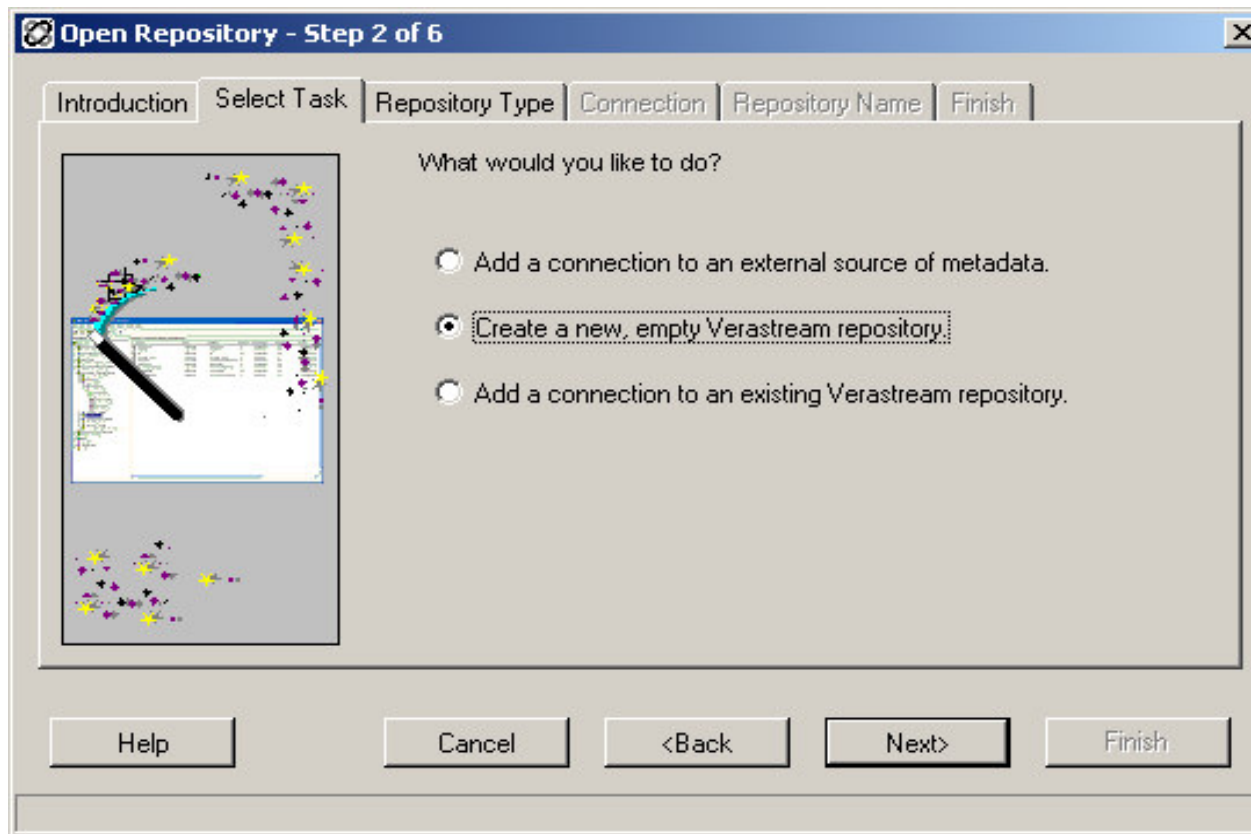
- Go to the Repository Manager
- Select the menu File → Open Repository



Building the Shipping Component

Create a Verastream Repository

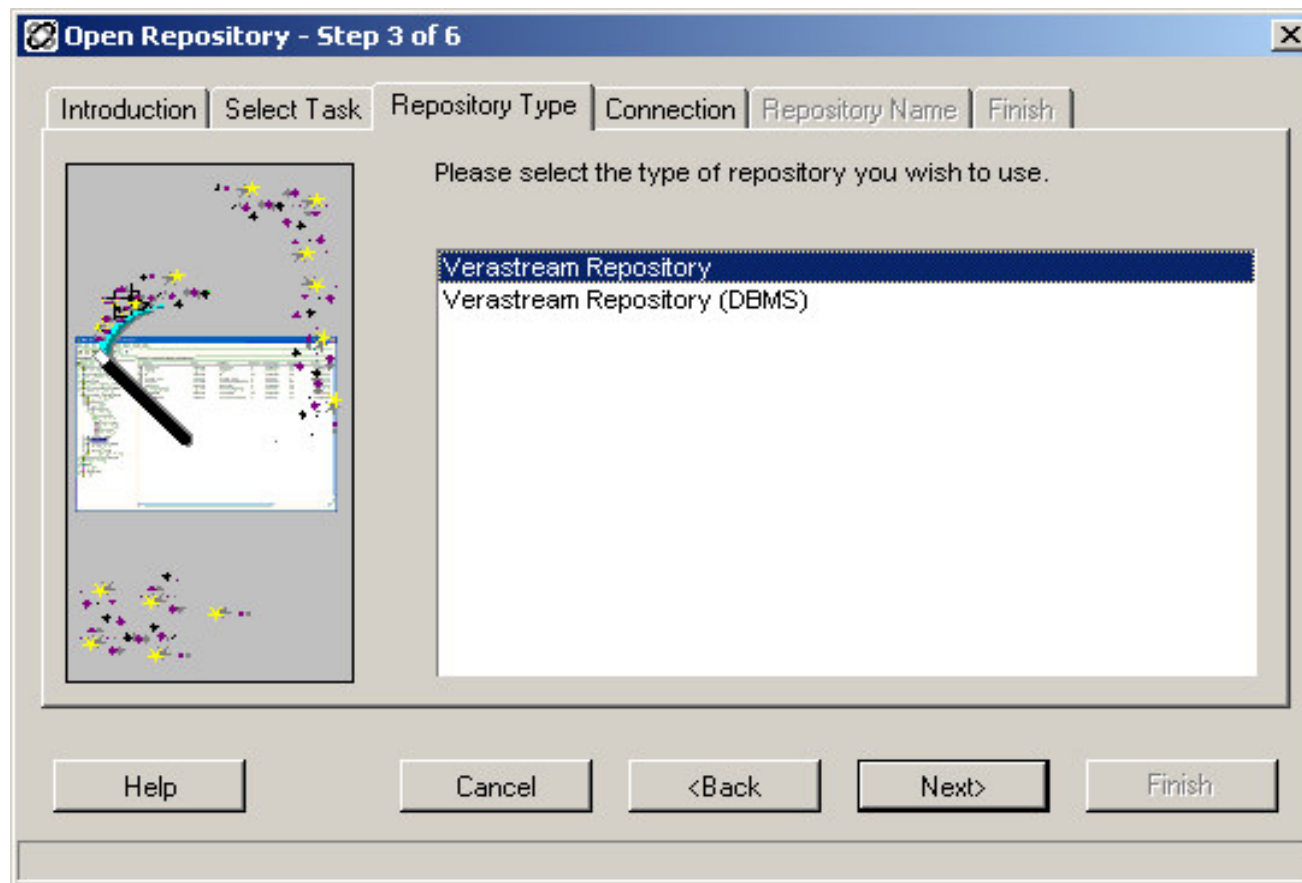
- Create a new, empty Verastream Repository



Building the Shipping Component

Create a Verastream Repository

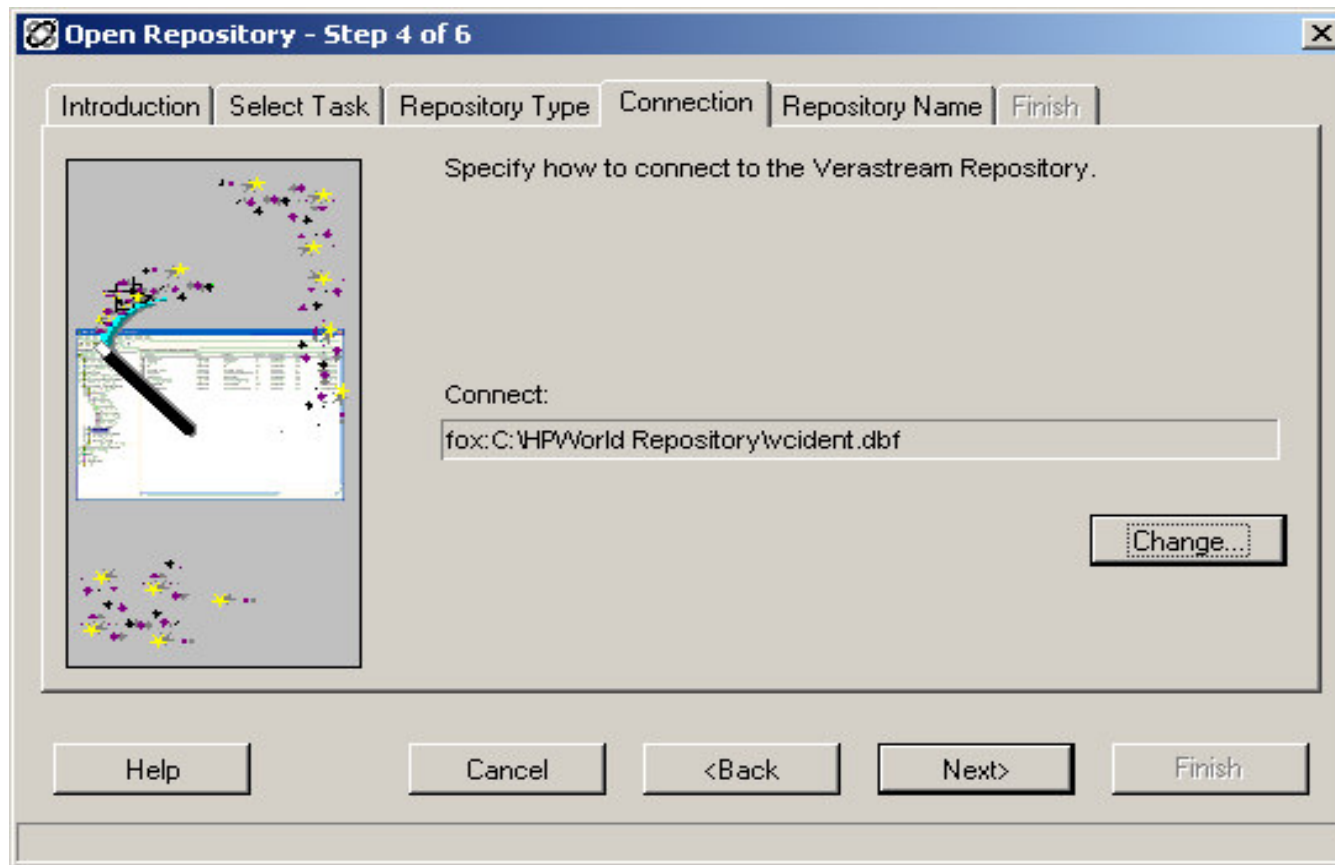
- Select Verastream Repository



Building the Shipping Component

Create a Verastream Repository

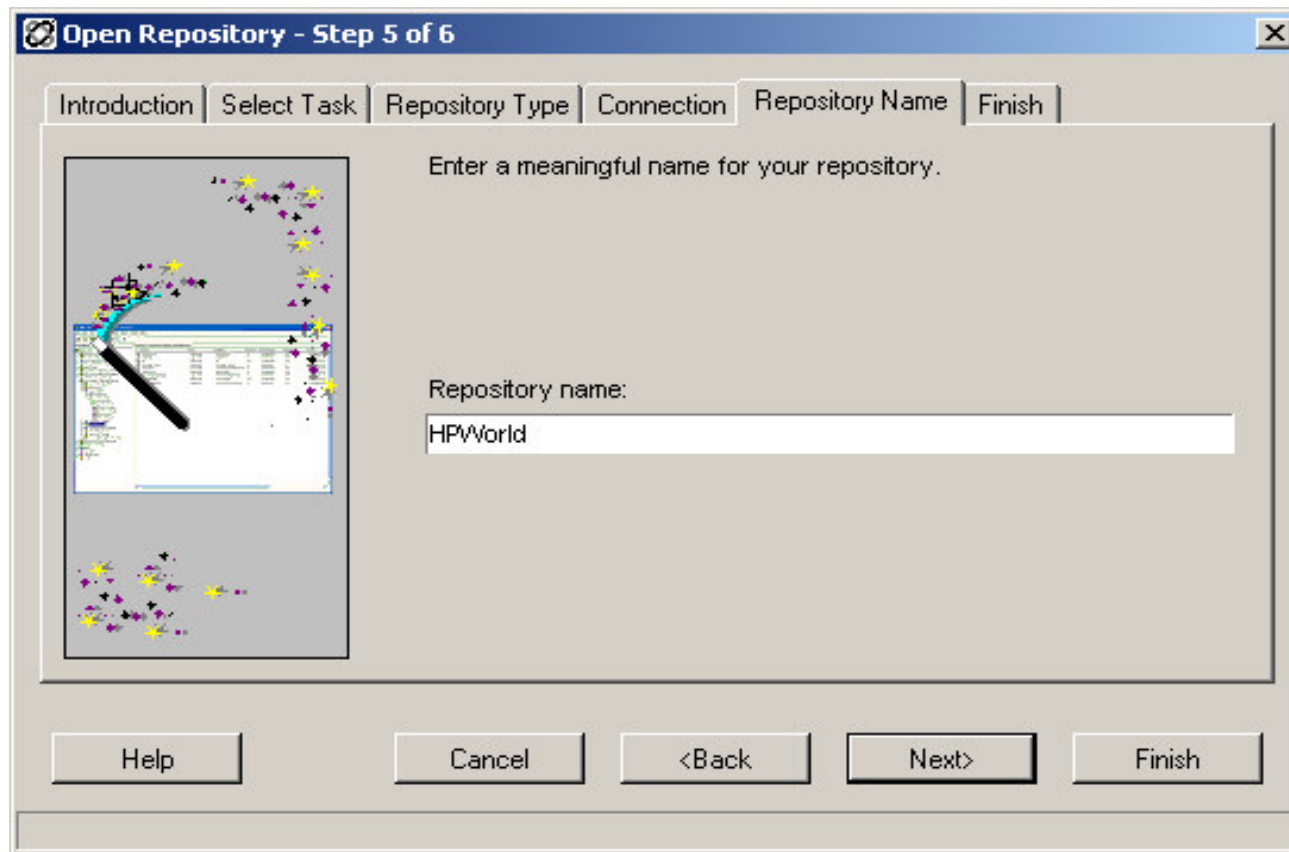
- Specify a directory



Building the Shipping Component

Create a Verastream Repository

- Name the Repository
- Click Finish



Build the components Verastream Shipping Component – Composite Application

- ☒ Open a Table Repository (To import metadata of RMS data files)
- ☒ Open a Verastream Host Integrator Model Repository
- ☒ Open a Java Repository
- ☒ Create a Verastream Repository
- ☐ Define the Shipping Component
- ☐ Create “claimsdd” module file
- ☐ Realize (Implement) the Component
- ☐ Register (deploy) the Component
- ☐ Test the Component

Define the Shipping Component

- Go to Repository Manager
- Select the Verastream Repository(HPWorld)
- Right click and select New → Component
- Follow instructions in Section I of ShippingComponent_Instructions.pdf file

Build the components Verastream Shipping Component – Composite Application

- ☒ Open a Table Repository (To import metadata of RMS data files)
- ☒ Open a Verastream Host Integrator Model Repository
- ☒ Open a Java Repository
- ☒ Create a Verastream Repository
- ☒ Define the Shipping Component
- ☐ Create “claimsdd” module file
- ☐ Realize(Implement) the Component
- ☐ Register(deploy) the Component
- ☐ Test the Component

Create “claimsdd” module file

- Follow instructions in Section II of ShippingComponent_Instructions.pdf file

Build the components Verastream Shipping Component – Composite Application

- ☒ Open a Table Repository (To import metadata of RMS data files)
- ☒ Open a Verastream Host Integrator Model Repository
- ☒ Open a Java Repository
- ☒ Create a Verastream Repository
- ☒ Define the Shipping Component
- ☒ Create “claimsdd” module file
- ☐ Realize (Implement) the Component
- ☐ Register (deploy) the Component
- ☐ Test the Component

Realize the Shipping Component

- Follow instructions in Section III of ShippingComponent_Instructions.pdf file

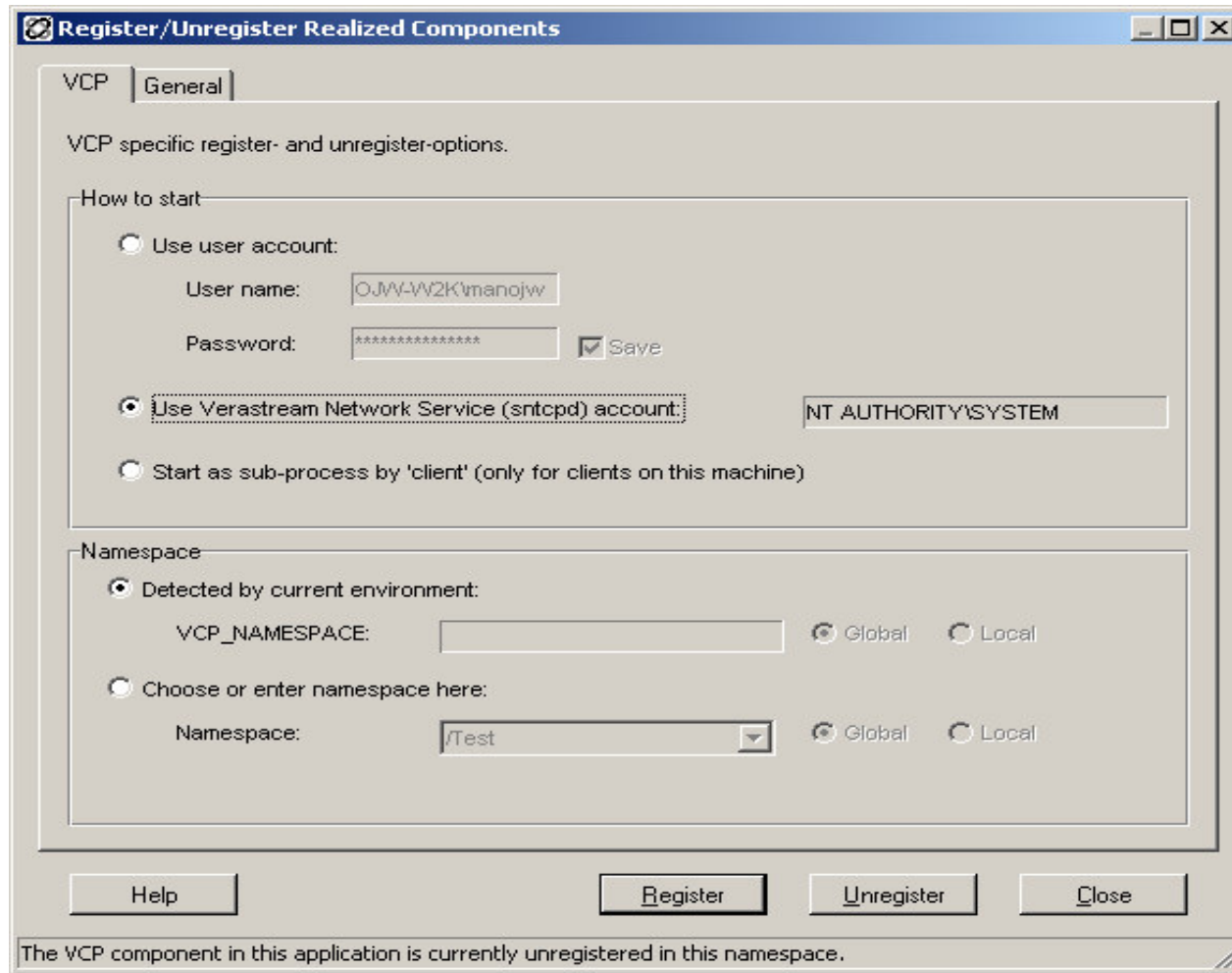
Build the components Verastream Shipping Component – Composite Application

- ☒ Open a Table Repository (To import metadata of RMS data files)
- ☒ Open a Verastream Host Integrator Model Repository
- ☒ Open a Java Repository
- ☒ Create a Verastream Repository
- ☒ Define the Shipping Component
- ☒ Create “claimsdd” module file
- ☒ Realize(Implement) the Component
- ☐ Register(deploy) the Component
- ☐ Test the Component

Deploy the Shipping Component

- Launch Process Integrator and open the shipping.lgc file
- Select the menu Register → Run → Register/Unregister Realized Components ...
- In the Register/Unregister Realized Components window, select the radiobuttons “Use Verastream Network Service(sntcpd) account and “Detected by current environment”
- Click the “Register” button

Deploy the Shipping Component



The image shows a Windows-style dialog box titled "Register/Unregister Realized Components". It has a "VCP" tab and a "General" sub-tab. The main area is labeled "VCP specific register- and unregister-options." and is divided into two sections: "How to start" and "Namespace".

How to start:

- ☐ Use user account:
 - User name:
 - Password: ☒ Save
- ☒ Use Verastream Network Service (sntcpd) account:
- ☐ Start as sub-process by 'client' (only for clients on this machine)

Namespace:

- ☒ Detected by current environment:
 - VCP_NAMESPACE:
 - ☒ Global ☐ Local
- ☐ Choose or enter namespace here:
 - Namespace:
 - ☒ Global ☐ Local

At the bottom, there are four buttons: "Help", "Register", "Unregister", and "Close". A status bar at the very bottom reads: "The VCP component in this application is currently unregistered in this namespace."

Build the components Verastream Shipping Component – Composite Application

- ☒ Open a Table Repository (To import metadata of RMS data files)
- ☒ Open a Verastream Host Integrator Model Repository
- ☒ Open a Java Repository
- ☒ Create a Verastream Repository
- ☒ Define the Shipping Component
- ☒ Create “claimsdd” module file
- ☒ Realize (Implement) the Component
- ☒ Register (deploy) the Component
- ☐ Test the Component

Test the Component

- Follow instructions in Section IV of ShippingComponent_Instructions.pdf file

Build the components Verastream Shipping Component – Composite Application

- ☑ Open a Table Repository (To import metadata of RMS data files)
- ☑ Open a Verastream Host Integrator Model Repository
- ☑ Open a Java Repository
- ☑ Create a Verastream Repository
- ☑ Define the Shipping Component
- ☑ Create “claimsdd” module file
- ☑ Realize (Implement) the Component
- ☑ Register (deploy) the Component
- ☑ Test the Component

“ We had numerous and complex demands, ranging from very high performance requirements to the fundamental need to retain our OpenVMS-based systems. WRQ was the only vendor with products and services that met or exceeded these demands, WRQ provides us with significant value, not only enabling us to modernize our infrastructure and applications but also to maximize the value of our OpenVMS systems.”

Dave Robinson

Vice President, MIS, Southeastern Freight Lines



HP WORLD 2004

Solutions and Technology Conference & Expo

Co-produced by:

