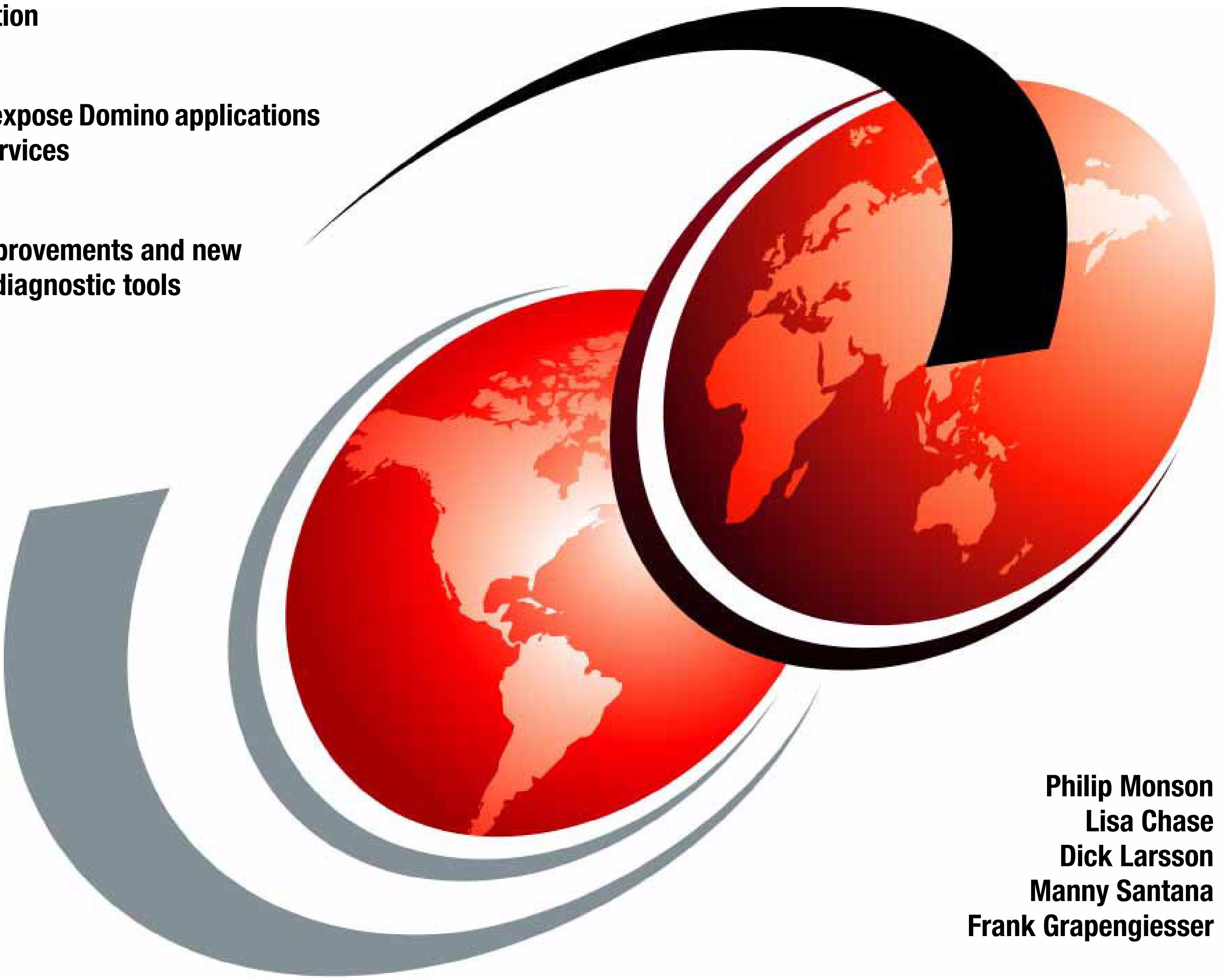
Lotus. software

# Lotus Domino 7 Application Development

Add storage and relational capabilities with DB2 integration

Extend and expose Domino applications with Web services

Usability improvements and new design and diagnostic tools



Philip Monson
Lisa Chase
Dick Larsson
Manny Santana
Frank Grapengiesser

Redpaper

ibm.com/redbooks

International Technical Support Organization

# Lotus Domino 7 Application Development

March 2006

**Note:** Before using this information and the product it supports, read the information in “Notices” on page vii.

**First Edition (March 2006)**

This edition applies to Version 7 of IBM Lotus Notes and Domino and Version 7 of IBM Lotus Domino Designer.

# Contents

# Notices

# Trademarks

# Preface

With Version 7, IBM® Lotus® Domino® Designer® software builds on its reputation for being a premier collaborative application development tool for Lotus Domino software-based applications. As an integral part of the IBM Workplace™ family, Lotus Domino Designer software helps companies improve employee productivity and build and deploy Lotus Domino applications more quickly, thereby enabling organizations to be more responsive to changing business initiatives. New features focus on tighter integration with Web standards, more interoperability with IBM technologies and ease of use.

This IBM Redpaper is designed to show application developers how to use the newest features in IBM Lotus Notes® and IBM Lotus Domino 7. This Redpaper takes you through powerful application development features, such as IBM DB2® Universal Database™ integration and Web services, by showing how a developer might enhance an example application for a fictitious company.

## The team that wrote this Redpaper

This Redpaper was produced by a team of specialists from around the world working at the International Technical Support Organization, Cambridge Center.

**Philip Monson** is a Project Leader at the ITSO Lotus Center in Cambridge, MA. Phil has been with Lotus and IBM for 15 years, joining the company when the early versions of Lotus Notes were rolled out for internal use. He has served in management, technical, and consulting roles in the IT, Sales, and Development organizations.

**Lisa Chase** is an engineer in the Lotus Software division of the IBM Software Group. She has been part of the IBM Messaging and Collaborative Software group for the past four years, has been a member of the Lotus Notes Client team, and is currently part of the team engineering the Domino Access for Microsoft® Outlook® client. Drawing on her knowledge of the Notes Client Calendar and Scheduling functionality and of Microsoft Outlook and Exchange, she is responsible for the current incarnation of the Domino Migration Tool.

**Dick Larsson** is Vice President of development and research at Ekakan (http://www.ekakan.com), a firm that assists enterprises to improve and develop collaborative solutions and knowledge management. His extensive experience with Domino and J2EE™ and intuitive grasp of new technologies have generated much acclaim within the European developer community. Prior to joining Ekakan, Mr. Larsson held various senior development positions at leading Swedish Lotus Notes consultancies. Mr. Larsson can be reached at mailto:dick.larsson@ekakan.com.

**Manny Santana** is an IT Specialist with IBM Global Services in Australia and is based in Sydney. He has more than 10 years of experience in development, deployment, and support of various internal IBM applications in the IBM Global Web Architecture and IBM Global Notes Architecture environments. He received a Bachelor of Science degree from the University of Sydney and is an IBM Certified Advanced Application Developer - Lotus Notes and Domino 6/6.5 and 7.

**Frank Grapengiesser** has been working as a Software Engineer for IBM Lotus customer support in Dublin for more than five years. Currently, he supports customers in the areas of Lotus Workflow™ and database design. Before that, he worked for two and a half years for the Bechtle AG in Mannheim, Germany developing Notes applications. He received a Diplom-Ingineur in mechanical engineering from the University of Hanover.

Thanks to the following people for their contributions to this project:

Mark Jourdain, Product Manager - Application Development, IBM Software Group, Lotus

Andre Guirard, Notes Client Team, IBM Software Group, Lotus

John Curtis, Senior Technical Staff Member - NSF/DB2 Transparency, IBM Software Group, Lotus

John Grosjean, Domino back-end classes - Web Services, IBM Software Group, Lotus

Steve Nikopoulos, Domino Programmability, IBM Software Group, Lotus

Jane L. Wilson, Knowledge System Architect, IBM Software Group, Lotus

Mary Jrolf, Advisory Software Engineer, IBM Software Group, Lotus

Russell Butek, Web Services Consultant, IBM Software Group, Application and Integration

Gary Rheaume, Senior Software Engineer, IBM Software Group, Lotus

Jennifer Dunne, IT Specialist, IBM Software Group, Lotus

Tom McGary, Senior Software Engineer, IBM Software Group, Lotus

## Become a published author

## Comments welcome

1

# An introduction to Lotus Domino 7 application development

Lotus Domino 7 application development represents a significant step forward in functionality that enables the user to create more powerful, easily distributable, and scalable IBM Lotus Domino applications. With this release, IBM Lotus Domino Designer offers application developers greater data storage and relational capabilities with the advent of IBM DB2 Universal Database (UDB) integration with Lotus Domino, the strength of increased data access and manipulation with Web services support, additional application design elements, and enhanced application debugging and diagnostic tools. Lotus Notes and Domino 7 offer developers the tools to create Lotus Domino applications that meet their unique business needs while presenting Lotus Domino application consumers with a professional and easy-to-use interface, whether accessed through a Lotus Notes, Lotus Domino for Web Access, or Internet browser interface. Additionally, the Web services support introduced in Lotus Notes and Domino 7 enables Microsoft .NET, Java™, Java 2 Platform, Enterprise Edition (J2EE), and other applications to interact with Lotus Domino without custom coding.

In this Redpaper, we demonstrate the new features and elements of the Lotus Domino 7 application development environment through a sample application we enhanced for ITSO Electronics, a fictitious corporation (consider the Redpaper team as consultants to the ITSO Electronics company). We show the application developer how to design a Lotus Domino application that uses the relational and storage capabilities of DB2 integrated with Lotus Domino, and then how to enhance the company's ability to deal with its external suppliers and partners by integrating Web services in the application, all the while using the design elements new to Lotus Domino 7. Finally, we offer instructions for using the new LotusScript and Java debugging features and profiling capabilities to debug and diagnose problems or performance issues.

## 1.1 The ITSO Electronics scenario

As previously mentioned, the following chapters of this paper discuss the new key application development features in Lotus Domino 7 in greater detail. To facilitate this discussion, a sample application is used throughout the book to provide examples of how to implement each new feature in the context of the ITSO Electronics business. This sample application is available for download from the IBM Redbooks Web site. For information about how to download this sample application, see Appendix A, “Additional material” on page 109.

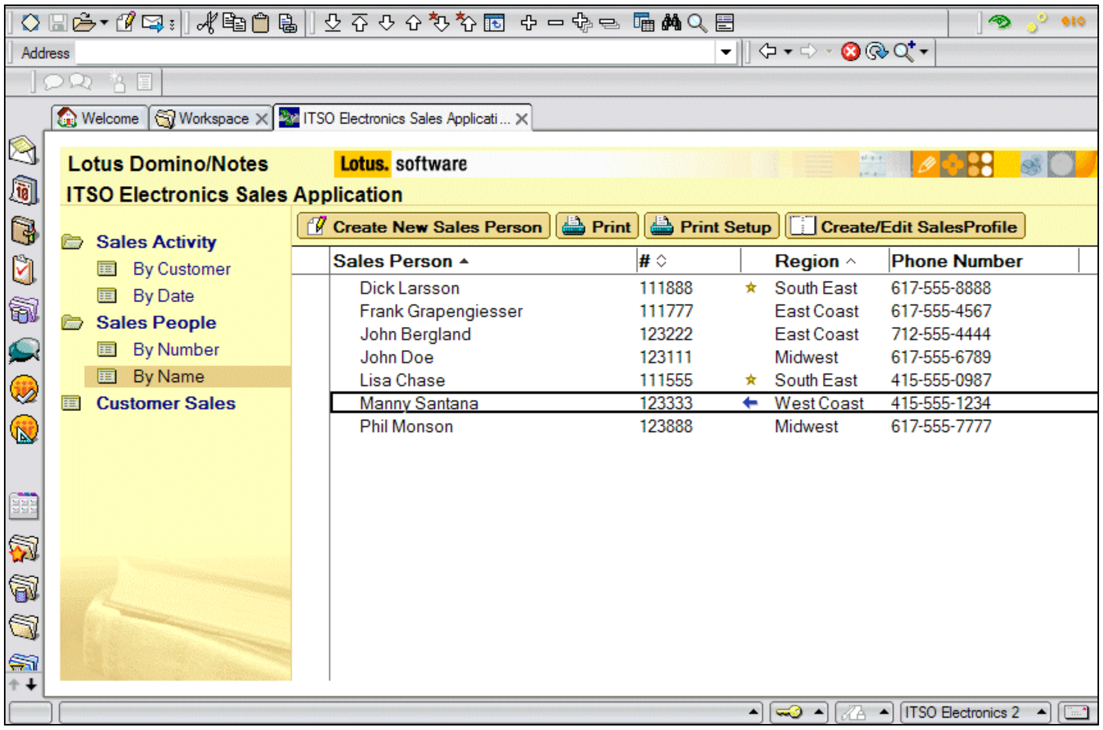
**Note:** This application is intended to provide examples of using the new application development features of Lotus Domino 7. It is not intended for use in a production setting.

### Overview of the ITSO Electronics application

The fictitious company ITSO Electronics is using Lotus Domino to support the sales department. Over the years, they have built a sales tracking application to help their sales force track interactions with their customers. The databases can be accessed from a Lotus Notes client or a Web browser.

There are two main databases in the ITSO Electronics sales tracking application: the Sales database and the Customers database. Figure 1-1 and Figure 1-2 on page 3 illustrate the user interface when accessed from Lotus Notes. A third database, Products, is used for product keywords and is not accessed directly by the sales force. Lastly, a fourth database has been added to the existing application environment for Web services elements.

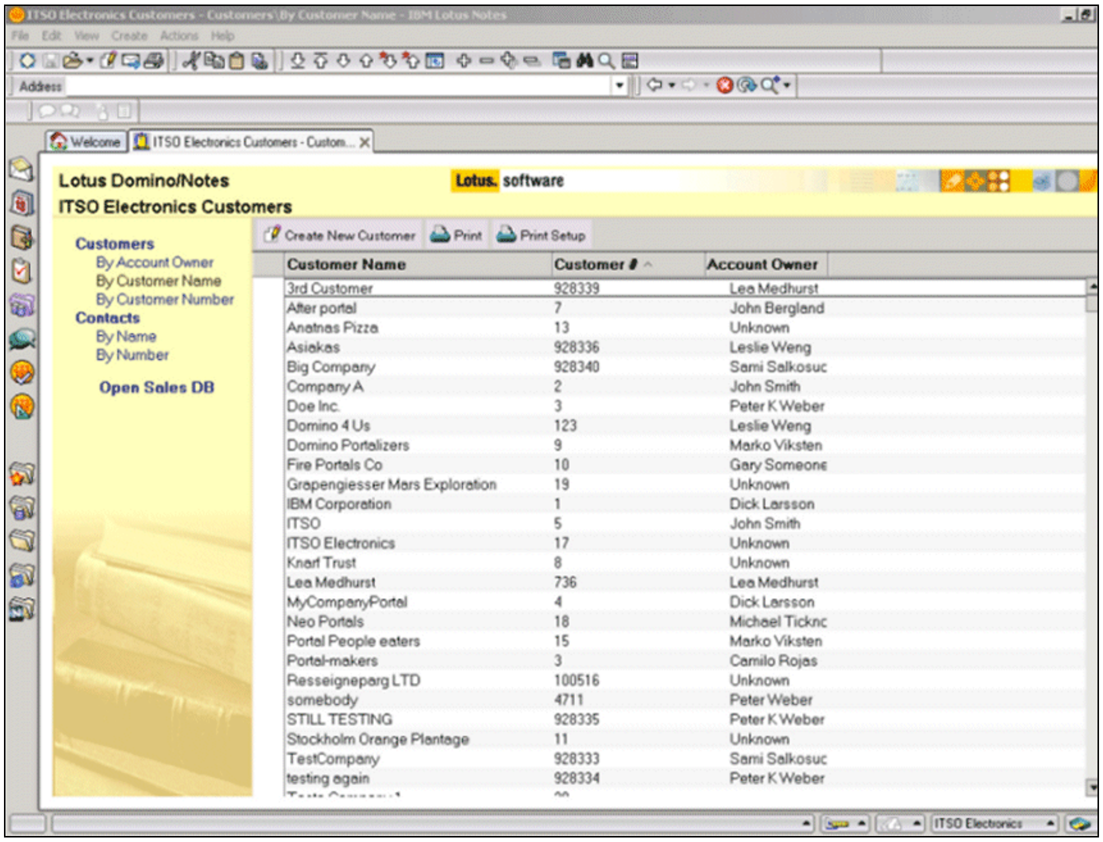
**Note:** All of the example scenario databases containing the new features described in this paper are included in Appendix A, “Additional material” on page 109 for your learning convenience.



| Sales Person | # | | Region | Phone Number |
|---|---|---|---|---|
| Dick Larsson | 111888 | ★ | South East | 617-555-8888 |
| Frank Grapengiesser | 111777 | | East Coast | 617-555-4567 |
| John Bergland | 123222 | | East Coast | 712-555-4444 |
| John Doe | 123111 | | Midwest | 617-555-6789 |
| Lisa Chase | 111555 | ★ | South East | 415-555-0987 |
| Manny Santana | 123333 | ← | West Coast | 415-555-1234 |
| Phil Monson | 123888 | | Midwest | 617-555-7777 |

*Figure 1-1 The Lotus Notes interface for the sales tracking application Sales database*

**Note:** Implementing Web services does not require the use of a separate Lotus Domino database, because Web services can be added to or incorporated in any Domino database. For the demonstration purposes of this paper, we used a separate database for simplicity and isolation of Web services information.



| Customer Name | Customer # | Account Owner |
|---|---|---|
| 3rd Customer | 928339 | Lea Medhurst |
| After portal | 7 | John Bergland |
| Anatnas Pizza | 13 | Unknown |
| Asiakas | 928336 | Leslie Weng |
| Big Company | 928340 | Sami Salkosuc |
| Company A | 2 | John Smith |
| Doe Inc. | 3 | Peter K Weber |
| Domino 4 Us | 123 | Leslie Weng |
| Domino Portalizers | 9 | Marko Viksten |
| Fire Portals Co | 10 | Gary Someone |
| Grapengiesser Mars Exploration | 19 | Unknown |
| IBM Corporation | 1 | Dick Larsson |
| ITSO | 5 | John Smith |
| ITSO Electronics | 17 | Unknown |
| Knarf Trust | 8 | Unknown |
| Lea Medhurst | 736 | Lea Medhurst |
| MyCompanyPortal | 4 | Dick Larsson |
| Neo Portals | 18 | Michael Tícknc |
| Portal People eaters | 15 | Marko Viksten |
| Portal-makers | 3 | Camilo Rojas |
| Resseigneparg LTD | 100516 | Unknown |
| somebody | 4711 | Peter Weber |
| STILL TESTING | 928335 | Peter K Weber |
| Stockholm Orange Plantage | 11 | Unknown |
| TestCompany | 928333 | Sami Salkosuc |
| testing again | 928334 | Peter K Weber |

*Figure 1-2 The Lotus Notes interface for the sales tracking application Customers database*

Figure 1-3 illustrates the architecture of the ITSO Electronics application, where the Sales, Customers, Products, and Web Services databases reside on the DB2 UDB server, but are accessed through the Lotus Domino server.

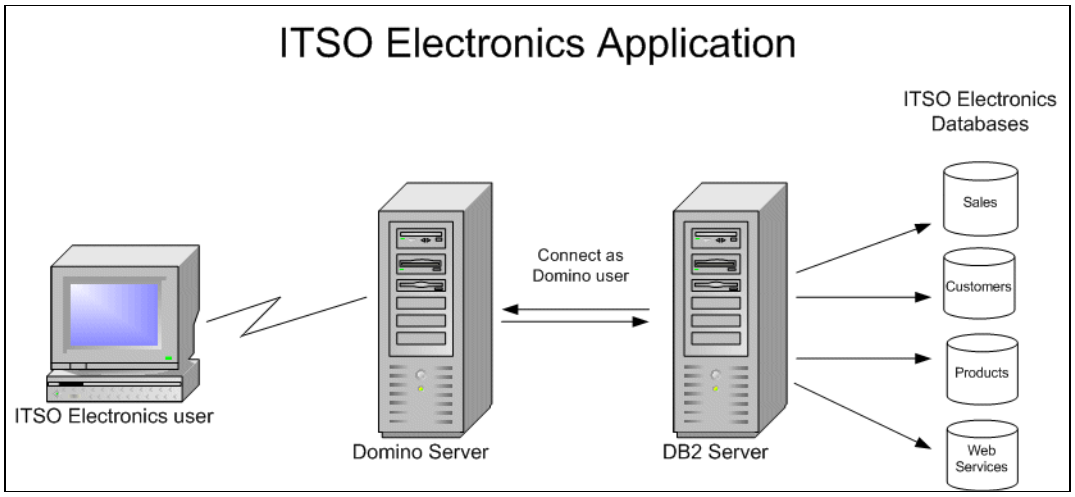


*Figure 1-3 Architecture of the ITSO Electronics application*

## 1.2 How this paper is organized

We present the development of the ITSO Electronics application in Lotus Domino 7 in the four chapters subsequent to this one.

In Chapter 2, “Using DB2 integration to enhance the ITSO Electronics application” on page 7, we explore the expansion of DB2 integration with Lotus Domino, and how DB2 data storage and relational capability can make the ITSO Electronics, and ultimately your, Lotus Domino application significantly more powerful. In contrast to historical relational database integration capabilities using Lotus Enterprise Integrator®, we examine the advantages of the DB2 architecture in Lotus Domino application development. We discuss how to prepare for DB2 and Lotus Domino integration, DB2 Access Views and Query Views, DB2 federation, and security considerations of integrating DB2 data storage with Lotus Domino and include troubleshooting tips for installing and configuring the DB2 environment.

Web services is perhaps the most exciting addition to the Lotus Domino application development toolkit, and in Chapter 3, “Enhancing the ITSO Electronics application with Web services” on page 43, we discuss the Web services technology and how it enhances the power and reach of the ITSO Electronics Lotus Domino application. While COM integration offered open access to Lotus Domino beginning in the 5.0x release, Web services moves beyond the constraints of the data access model to offer Lotus Domino developers data services integration. We examine service-oriented architecture (SOA) and how Web services fits into the SOA scheme, as well as Web Services Description Language (WSDL), Simple Object Access Protocol (SOAP), the process for adding the Web services element to a Lotus Domino application, exception and error handling, security, and consuming Web services.

Lotus Domino Designer 7 contains several powerful new design elements, which we discuss and demonstrate in Chapter 4, “Implementing new design elements in the ITSO Electronics application” on page 77, as we enhance the ITSO Electronics application. We examine additions to the formula language, including LotusScript and Java classes, as well as right-click actions and new shared column capabilities, in detail.

In Chapter 5, “Diagnosing and troubleshooting the ITSO Electronics application” on page 95, we cover the new diagnostic and debugging tools in the Lotus Domino 7 application development environment. Using the ITSO Electronics application as an example, we examine the power of code profiling to optimize your Lotus Notes application, the additions to

LotusScript debugging capabilities, and new Java debugging capabilities, including remote Java debugging.

## 1.3 Application development in Lotus Domino Designer

IBM Lotus Domino Designer is a powerful application development environment for creating business applications that use a Lotus Domino server. For the purposes of this paper, we assume that you are an experienced Lotus Domino application developer familiar with the application development features and Lotus Domino Designer design elements in Lotus Domino Release 6. Our demonstration of the ITSO Electronics sample application development begins with the addition of DB2 as a data storage and relational element, assuming you are well versed in how to build a Lotus Domino application without the DB2 component. If you are not familiar with the Lotus Domino Designer application development environment, refer to the IBM Redbook *Domino Designer 6: A Developer's Handbook*, SG24-6854, and familiarize yourself with the Lotus Domino Designer design elements and features, available at:

`http://www.redbooks.ibm.com/abstracts/sg246854.html`

2

# Using DB2 integration to enhance the ITSO Electronics application

In this chapter, we discuss the new functionality enabling the integration of IBM Lotus Notes and Domino 7 with IBM DB2 Universal Database (UDB). Wherever appropriate, we use the ITSO Electronics application as an example to explain the new design features.

On the pages that follow, you will find information about the following topics:

- Introducing Domino and DB2 integration
- DB2 Access Views
- Query Views
- Federated DB2 data from DB2
- Security considerations regarding DB2 and Lotus Domino
- Troubleshooting DB2 and Lotus Domino

**Note:** IBM Lotus Notes and Domino 7 with DB2 is released with Limited Availability; as such, it is not available for general usage, but is provided for evaluation purposes only. For more information about the Limited Availability program and other important information about this topic, see:

`http://www.ibm.com/lotus/nsfdb2`

## 2.1 Introducing Domino and DB2 integration

Lotus Notes and Domino have a proven track record of integration with relational databases, transactional and enterprise resource planning (ERP) systems through features and products such as Lotus Enterprise Integrator, Domino Enterprise Connection Services, Data Connection Resources, and Lotus Connectors LotusScript Extensions. Lotus continues these features and products with Notes and Domino 7 and provides new non-programmatic, decentralized application development options for integration with IBM DB2 relational databases that can be easily used by line-of-business users and senior developers alike. The Lotus and DB2 brands within IBM have worked together to refine their integration capabilities, and the Domino 7 server provides the ability to selectively specify new or existing Domino databases as DB2 enabled. DB2-enabled Domino databases are accessed by users transparently through the Domino 7 server but are stored on the DB2 system. This allows the DB2-enabled Domino database to use relational constructs and allow data from the DB2-enabled Domino database to be used in relational constructs, while continuing to support the Notes API features, including replication, mail routing, and administration access. Because IBM DB2 is a relational database system, data from separate DB2 databases, tables, views, and federated objects can easily combined into new data sets using their common data fields and relationships. This enables developers to work with and present data in new ways and facilitates the creation of more integrated and valuable Domino and non-Domino applications that use Domino data, DB2 data, and relational capabilities. We discuss these powerful new capabilities in greater detail later in this chapter.

Before using the new Domino and DB2 integration features, you must first install a DB2 server and configure your Domino 7 server to connect to the DB2 server. In Lotus Domino 7, the new DB2 integration features are available on the Microsoft Windows® and IBM AIX® 5L™ operating systems.

### Local DB2 installation

When setting up the Domino and DB2 integration, you can choose between a local or remote installation. In the location installation type, both the Domino server and DB2 server are installed on the same machine. A local installation is the choice for a test environment or a small production system. Figure 2-1 shows a system with a local installation.

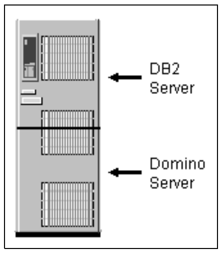


*Figure 2-1 Local installation: Domino server and DB2 server installed on the same machine*

### Remote DB2 installation

For larger installations or systems with a high server load, you will probably opt for the remote installation. Here, the Domino server and the DB2 server are on different machines. See Figure 2-2 on page 9 for an example of a remote installation.

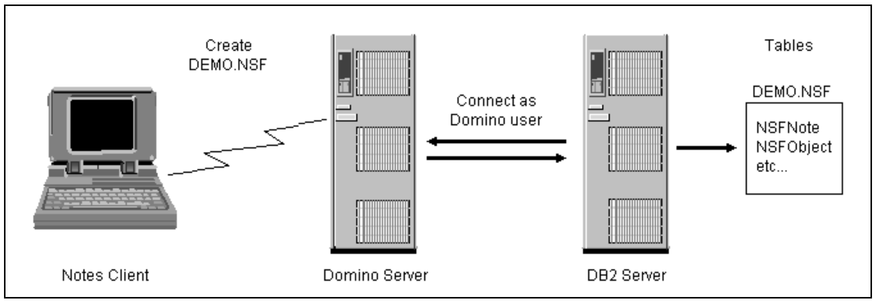


*Figure 2-2 Remote installation: Domino server and DB2 server installed on different machines*

There are two levels of DB2 integration. In the first scenario, the Domino server and DB2 server are installed on the same machine, and in the second scenario, the Domino server and DB2 server are installed on separate machines.

If you install both the Domino and DB2 servers on the same machine, follow this procedure:

1. Install Domino 7 on the Domino server.
2. Install these elements on the DB2 server:
   a. DB2 UDB Enterprise Server Edition Version 8.2.2, including these files:
      i. DB2 UDB Enterprise Server V8.2
      ii. DB2 Universal Database Version 8 Fix Pack 9a
   b. Limited availability enablement key
3. Install the DB2 Access server.

If you install the Domino server and the DB2 server on different machines, follow this procedure:

1. Install Domino 7 on the Domino server.
2. Install these elements on the DB2 server:
   a. DB2 UDB Enterprise Server Edition Version 8.2.2, including these files:
      i. DB2 UDB Enterprise Server 8.2
      ii. DB2 Universal Database Version 8 Fix Pack 9a
   b. Limited availability enablement key
3. Install DB2 Run-Time Client on the Domino server.
4. Install the DB2 Access server on the same machine as the DB2 server.

**Note:** Make sure that you remove any versions earlier than 7.0 of Lotus Notes and Domino from the machine on which you install the DB2 Access server. Otherwise, your system will not work. If you need to keep the earlier Lotus Notes/Domino server, rename the program folder of these installations. Ensure that nnotes.dll is *not* in your PATH.

The focus of this paper is the new design features of Domino 7. Therefore, we only provide a high-level overview of the setup process. For detailed information about how to set up and maintain Domino and DB2 integration, refer to:

- *Lotus Domino Administrator 7 Help*, “Domino and DB2” topic

  `http://www.lotus.com/ldd/doc/domino_notes/7.0/help7_admin.nsf/`
- Readme70.pdf (on the CD)

- DB2 learning resources from the IBM training and certification site

  http://www.ibm.com/software/sw-lotus/services/education.nsf/wdocs/educationhomepage

  – Developing Applications with IBM Lotus Domino Enabled for DB2, E-learning Edition: Self Paced (N7D400SP)

    http://www.ibm.com/software/sw-lotus/services/education.nsf/81F04813DC9CE3C7852566DA004C5CB2/A6DF60493B0D2B13852570A7006C7E9B

  – Introducing IBM DB2 Concepts (N7D020)

    https://education.lotus.com/rw/lewwschd.nsf/5FDC13DD71D1143185256575001E5133/E55E8003DCEC00FA8525701A006959C4

**Tip:** During the setup, you have to set a number of DB2 user names. DB2 user names should conform to the following DB2 naming rules and should always be uppercase:

- User IDs on Linux® and UNIX®-based systems can contain up to eight characters.
- User names on Microsoft Windows can contain up to 30 characters. Microsoft Windows NT®, Windows 2000, Windows XP, and Windows Server 2003 currently have a practical limit of 20 characters.
- When not using client authentication, non-Windows 32-bit clients connecting to Windows NT, Windows 2000, Windows XP, and Windows Server 2003 with user names longer than eight characters are supported when the user name and password are specified explicitly.
- Names and IDs cannot:
  – Be USERS, ADMINS, GUESTS, PUBLIC, LOCAL, or any SQL reserved word
  – Begin with IBM, SQL, or SYS
  – Include accented characters

## 2.2 DB2 Access Views

Lotus Notes and Domino use a completely different storage concept than relational databases such as DB2. While in relational databases the information is organized in well-defined tables, the Domino database stores documents or “notes” composed of a list of fields in a flat-file format. This unstructured way of storing information makes Domino data difficult to access from a DB2 perspective. To resolve this problem, Domino 7 introduces DB2 Access Views, shared resources that let you define DB2 views of Domino data, organizing the data in a table that can be queried with Structured Query Language (SQL). DB2 Access Views are background design elements and are not visible to end users.

DB2 Access Views are only available for databases stored on a DB2 server. Before creating a DB2 Access View, your Domino server must be DB2 enabled and your database has to be hosted on the DB2 system. If you create a local replica of your database or replicate your database on a non-DB2-enabled Domino 7.0 server, the DB2 Access View will still be in the database but will not be usable or visible in Lotus Domino Designer.

When a Lotus Notes database is enabled for DB2, Domino creates a DB2 database schema to hold any user-accessible views of the data in the Lotus Notes database data. The schema name is based on the NSF file name. Any DB2 Access Views you create will also be located in this schema.

When you create a DB2 Access View, the information is copied and redundantly stored in a DB2 table. After you have created and initialized the data in the DB2 Access View, Domino automatically synchronizes updates between the free-form Domino data and the structured column data in the DB2 Access View. Therefore, there is no need to refresh DB2 Access View.

## 2.2.1 Creating DB2 Access Views

To create a new DB2 Access View, follow these steps. Also refer to the Lotus Domino 7 Designer, available at:

http://www.lotus.com/ldd/doc/domino_notes/7.0/help7_designer.nsf

1. Open the DB2-enabled database in Lotus Domino Designer 7.0.
2. In the design pane, select **Shared Resources** → **DB2 Access views**.
3. Click the design action button **New DB2 Access view**. The window shown in Figure 2-3 opens.

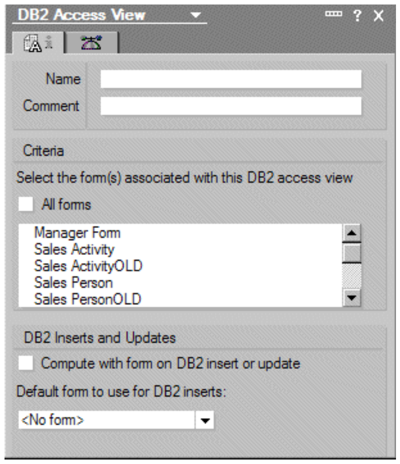


*Figure 2-3 Basic properties tab of DB2 Access View*

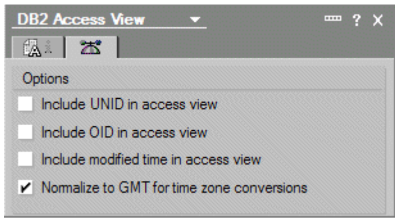
4. In the DB2 Access View window:
   - Assign a name to the DB2 Access View.
   - Select the form or forms associated with this DB2 Access View.

     You can select which documents are included based on the content of the "Form" field of the document. You can either select all forms or choose some of the available forms from a list. If a document does not have a form field, it will only be included in the DB2 Access View if you select **All forms**.
   - The DB2 Inserts and Updates settings are only valid if you run insert or update SQL statements from DB2:
     - Compute with form on DB2 insert or update

       Enable this option if the selected forms contain computed fields and you want the formulas to be computed when the note is created or updated using SQL.
     - Default form to use for DB2 inserts

       DB2 users can perform inserts, updates, and deletes (given the right permissions) against data in the DB2 Access View. For Domino 7.0, you can only choose a single type of document (based on FORM) to create using the SQL INSERT statement. Documents of any type (FORM) can be updated or deleted from SQL.

5. Switch to Advanced tab on the window, as shown in Figure 2-4.



Options

- [ ] Include UNID in access view
- [ ] Include OID in access view
- [ ] Include modified time in access view
- [x] Normalize to GMT for time zone conversions

*Figure 2-4 Advanced tab of DB2 Access View*

Here you can select the following options:

– Include UNID in access view

When you select this check box, you will be able to access the document unique ID of a Lotus Notes document through a DB2 Access View.

– Include OID in access view

OID stands for Origination Identifier. If you select this option and select it in the Query View's SQL formula, document links will work when view entries are selected and copied as a table.

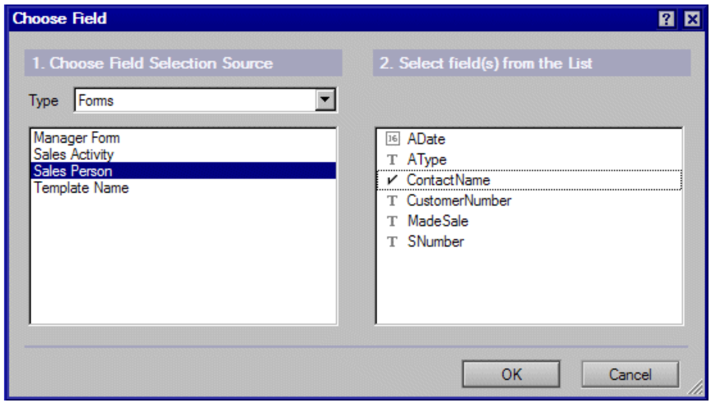
– Include modified time in access view

Select this option to include the Notes modified time for each note in the DB2 Access View. Note that this time stamp is modified to GMT.

– Normalize to GMT for time zone conversions

Select this option to standardize all dates and times in the DB2 view to GMT. This is especially useful for distributed DB2 applications that are accessed by users in different time zones.

6. Close the DB2 Access View properties window.
7. Select the **Choose Field** action, and the window shown in Figure 2-5 opens.


Choose Field
1. Choose Field Selection Source
2. Select field(s) from the List
OK
Cancel

Type: Forms

Manager Form
Sales Activity
Sales Person
Template Name

ADate
AType
✓ ContactName
CustomerNumber
MadeSale
SNumber

*Figure 2-5 DB2 Access View Choose Field window*

8. In the Choose Field window:
   a. Select the fields you want to include from each form.

      In Domino 7, a DB2 Access View can only have a maximum of 84 columns. If you need more data than this, you need to define multiple DB2 Access Views.

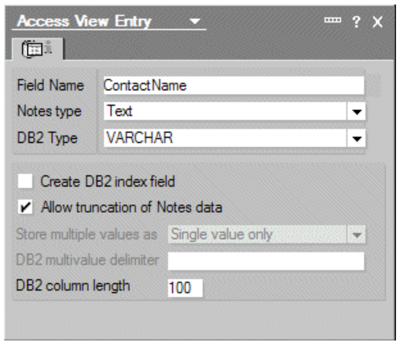
   b. Click **OK** to confirm your choices.

A list with the selected DB2 Access View entries appears in the Work pane.

9. Select the **Insert Field** action to include further fields.

   You might want to add further fields that are not on the selected forms. An example might be the Form field. You can manually add such field using the action **Insert Field**. Each time you click the action, an unnamed DB2 entry is added to the work pane.

10. Double-click each Access View Entry to edit its properties, as shown in Figure 2-6.



Field Name: ContactName
Notes type: Text
DB2 Type: VARCHAR
Create DB2 index field: ☐
Allow truncation of Notes data: ✓
Store multiple values as: Single value only
DB2 multivalue delimiter:
DB2 column length: 100

*Figure 2-6 Access View Entry properties*

The first three properties should be populated for fields added by the Insert Field action. For fields selected from forms, appropriate properties are populated automatically. This window has the following fields:

– Field Name

  This is the matching field name for the DB2 column in the DB2 Access View, so it must be the actual name of a field that appears in the note (otherwise, no matching field would be found and the column in the DB2 view would be blank).

  **Note:** If you used the Choose Field method of specifying the fields for the DB2 Access View, you will be able to edit the field names in the properties. However, if you change the field name so that it no longer matches a field name in the note, the corresponding column in the DB2 will be blank.

– Notes type

  Indicate the Lotus Notes data type for this field.

  **Note:** Formula, rich text, and rich text light Notes data types are not supported for use in DB2 Access Views.

– DB2 Type

  Indicate the DB2 data type for this field. Lotus Notes will indicate a default value associated with the Notes type you choose.

  **Note:** If you have an integer defined in the DB2 Access View, and you try to update that column with a real number (with a decimal) using the SQL INSERT or UPDATE statements, DB2 will convert the value to an integer before performing the update and the decimal value will be truncated.

The following attributes are relevant for all DB2 Access View entries:

- Create DB2 index field: Creates this column as a DB2 index field, which keys the database record for rapid retrieval. Using the following command, this option creates a single DB2 index on the column in the table:

  ```
  > CREATE INDEX ON TABLE x (coly)
  ```

  This does consume some space and requires more resources to update, but also makes queries like this run much faster, particularly because TABLE x will retrieve a lot of rows:

  ```
  > SELECT cola, colb, colc FROM x WHERE coly=<some value>
  ```

  Indexing in DB2 is an advanced topic for some Domino developers, but there are indexes that cannot (currently) be created from Domino Designer that optimize a wide variety of select statements. For example, this command optimizes the following query:

  ```
  > CREATE INDEX ON TABLE x (cola, colb, colc)
  ```

  In this example, the index is used to sort the data, which, without an index, is one of the most resource-consumptive operations you can do when tables get large:

  ```
  > SELECT cola, colb, colc FROM x WHERE cola = <some value> ORDER BY cola, colb, colc
  ```

- DB2 column length: You define the length of the column in the DB2 Access View with this property. When choosing the value for this field, you often will have to find a compromise between conflicting targets. You might want to display as much as possible of certain fields, but be aware that the maximum column length directly impacts the space the DB2 Access View fills on the DB2 database. When you map Notes fields, there is a limit to the total length of data you can save in the DAV table. The length of all data fields, plus accommodating for DB2, cannot exceed 16,384 bytes of data. If you need to store more than 16 k, you can map some of your text fields to LONG VARCHAR. When you map a field to LONG VARCHAR, a small "locator" is stored "inline" (counting toward the 16,384 byte limit) and the actual field data (up to 32,000 bytes per field) is stored as a character large object (CLOB). Using CLOBs gives a DAV more than enough storage capacity (84 * 32,000), but you should only use LONG VARCHAR if you cannot fit the data "inline," because CLOB storage is slower than inline storage, and CLOBs use more disk space than inline storage. When you validate or create a DB2 Access View, the Domino server will check if the DB2 Access View definition exceeds the maximum length, so you do not need to manually calculate the length. However, if wanted, you can use the guidelines in Table 2-1 to estimate the "inline" data length.

*Table 2-1 Data types and length in DB2 Access View*

| Type | Length in DAV | Comment |
|---|---|---|
| VARCHAR | 5 + declared length | For example, VARCHAR(40) requires 45 bytes of storage. |
| LONG VARCHAR | 25 | Inline locator. |
| DOUBLE | 9 | Fixed size. |
| TIMESTAMP | 11 | Fixed size. |
| DATE | 5 | Fixed size. |
| TIME | 4 | Fixed size. |
| INTEGER | 9 | Fixed size. |

– Allow truncation of Notes data: Sometimes, the size of a field will be larger than the maximum column length you defined for a DB2 Access View Entry. When you allow truncation, the content will be cut off when the maximum length has been reached. If you do not allow truncation, you will find the following message on the console log when the limit is reached while populating the DB2 Access View: “Access Table field data exceeds defined length.”

  In Domino 7, Allow Truncation of Notes data is disabled by default. As long as you use DB2 Access Views, we strongly recommend that you enable this property for every DB2 Access View entry against text data. There is one exception: If you plan to send SQL updates on DB2 Access Views from DB2, this setting gives permission to truncate the data to this length. This can lead to loss of data.

  **Important:** We recommend that you select the **Allow truncation of Notes data** option for all Access View entries.

## 2.2.2 Exporting TIMEDATES to DB2 Access Views

The Lotus Notes TIMEDATE can store the date, the time, the time zone, and a daylight savings time indicator, but a DB2 time stamp only stores the date and time. DB2 assumes that all dates and times are local to the DB2 server's time zone. This can cause the incorrect time to display when exporting TIMEDATEs to DB2 Access Views, or reading TIMESTAMPS from DB2 in a Query View (whether they are from DB2 Access Views or other DB2 tables).

There are two options for saving the date and time in DB2 Access Views: local time or modified to GMT (standardized). You choose the way you want them handled using the Advanced tab on the Access View Entry properties page, and then set the “Normalize to GMT for time zone conversions” field accordingly.

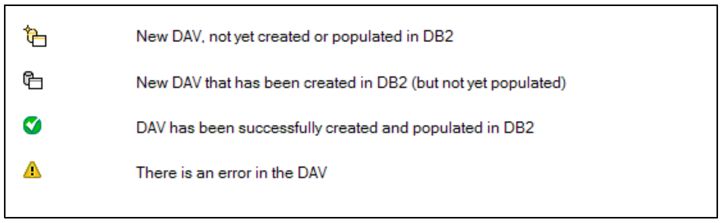
**Note:** You must rebuild your DB2 Access View if you change this setting for an existing DB2 Access View.

## 2.2.3 Inserting date ranges in a DB2 Access View

When inserting or updating date ranges in a note inside of a DB2 Access View through SQL, you must specify the date ranges with the full DB2 time stamp, such as yyyy-mm-dd-hh.mm.ss.subsec or 2005-12-12-08.06.30.123456. Failure to do so will result in the value being inserted as a null.

## 2.2.4 Checking the status of a DB2 Access View

Selecting the DB2 Access Views in Domino Designer shows a list of all the DB2 Access Views in the database. Before each DB2 Access View in the list, you find an ion that symbolizes the status of each DB2 Access View, as shown in Figure 2-7.



New DAV, not yet created or populated in DB2

New DAV that has been created in DB2 (but not yet populated)

✓ DAV has been successfully created and populated in DB2

⚠ There is an error in the DAV

*Figure 2-7 Icons symbolizing the status of a DB2 Access View*

## 2.2.5 Additional considerations when creating a DB2 Access View

You can create as many DB2 Access Views in the database as you want, for example, one for each form. However, it might be easier to create one or two large DB2 Access Views and include the data from all the forms you want to access and all the documents. To limit the number of columns in DB2 Access Views, try to use the same name for corresponding fields in different forms.

Domino is incredibly flexible when it comes to fields and their data types. Two documents based on the same form in Lotus Notes and Domino can have different data types. This advantage of flexibility can become a problem when you try to create DB2 Access Views. DB2 is very strict with respect to data types. It will often be necessary to clean the data in a database and ensure that all content of the fields used in DB2 Access Views have the same data type.

It is not a problem if a certain field is missing in a Lotus Notes document or if its value is empty. In the DB2 Access View, this field will simply be set to null and any queries will display an empty value. Rich text and formula fields are not supported in DB2 Access Views. If you include fields with one of this field types, you will not get error messages; however, the content will be displayed as null.

The size of a DB2 Access View and its entities (tables, views, triggers, and indexes) are not included in the total file size of the DB2-enabled Lotus Notes database for which it was created. Therefore, if you have numerous DB2 Access Views associated with a DB2-enabled Lotus Notes database, be aware that they occupy disk space above and beyond that of the DB2-enabled Lotus Notes database. However, concerns within IBM that DB2 Access Views take up large amounts of disk space have proved to be unfounded. Early internal deployments have shown that even a DB2 Access View with a large number of columns typically takes up only about 2 to 3% of the database space.

## 2.2.6 Implementing DB2 Access Views in the ITSO Electronics application

In our example scenario, ITSO Electronics wants to make use of the new Query Views feature, discussed in 2.3, “Query Views” on page 18. To prepare for the Query Views, we now create some DB2 Access Views. In this example, we decide to have one DB2 Access View in each of the Lotus Notes databases of the application.

First, we create the DB2 Access View SALESDAV in the ITSO Electronics Sales database. Table 2-2 shows the selected fields.

*Table 2-2 DB2 Access View SALESDAV in the Sales database*

| **Fields included in DB2 Access View** | **Form** |
|---|---|
| ANumber | Sales Activity |
| AType | Sales Activity |
| ContactName | Sales Activity |
| CustomerName | Sales Activity |
| CustomerNumber | Sales Activity |
| MadeSale | Sales Activity |
| Product1 | Sales Activity |
| Product2 | Sales Activity |

| Fields included in DB2 Access View | Form |
|---|---|
| Product3 | Sales Activity |
| Product4 | Sales Activity |
| SName | Sales Activity, Sales Person |
| SNumber | Sales Activity, Sales Person |
| SPhone | Sales Person |
| STitle | Sales Person |
| FORM | Sales Activity, Sales Person |

The next DB2 Access View to be created is the CUSTOMERDAV in ITSO Electronics Customer database. See Table 2-3.

*Table 2-3 DB2 Access View CUSTOMERDAV in the Customer database*

| Fields included in DB2 Access View | Form |
|---|---|
| customerAddress | Customer |
| CustomerDescription | Customer |
| customerName | Customer |
| customerNumber | Customer |
| ownerName | Customer |
| FORM | Customer |

Finally, we create the DB2 Access View PRODUCTSDAV in the ITSO Electronics Products database.

*Table 2-4 DB2 Access View PRODUCTSDAV in the Products database*

| Fields included in DB2 Access View | Form |
|---|---|
| productDescription | Product |
| SupplierAddress | Supplier |
| SupplierDescription | Product, Supplier |
| SupplierName | Supplier |
| SupplierNumber | Product, Supplier |
| productName | Product |
| productNumber | Product |
| FORM | Product, Supplier |

We make use of each of these DB2 Access Views when creating Query Views in 2.3, “Query Views” on page 18.

## 2.3 Query Views

Query Views are a new type of Notes database (NSF) view. They are populated by using queries written in the Structured Query Language (SQL). At first glance, Query Views are just another way to collect and display data. On closer inspection, Query Views transform the way you look at data in Lotus Domino. For the first time, it is possible to display information out of more than one Lotus Notes databases in one view. You can use the following SQL constructs:

- As Join

  Documents out of related entries in different DB2 Access Views are joined into one row in the Query View, as described in 2.3.2, “Creating a Query View using a join” on page 20.
- As Union

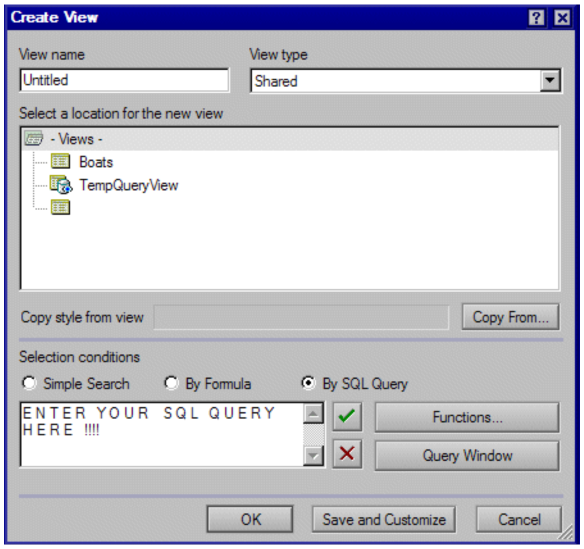
  Entries from different DB2 Access Views are assembled into one view. We explain this in 2.3.3, “Creating a Query View using a union” on page 21.
- Group by
- Having

The queries are dynamically built using SQL and are not taking space in the database. The SQL statement, in turn, can be dynamically built using Lotus Notes formula language. You you can freely use @formulas such as @Username and @Today without causing indexing problems as in conventional Lotus Notes views. You can even control the content of the view with user interactions such as @Promp. We describe this in 2.3.4, “Creating Query Views dynamically using @formulas” on page 22.

### 2.3.1 Creating a Query View

Query Views cannot directly display Lotus Domino document data. To display Lotus Domino data, you first need to create a DB2 Access View, as described in 2.2.1, “Creating DB2 Access Views” on page 11. After you have a DB2 Access View in place, you can start to create a Query View using the following steps:

1. In Domino Designer, go to **Views**. Click the **New View** action button. In the Create View window, select **By SQL Query** as the selection condition, as shown in Figure 2-8.



View name: Untitled
View type: Shared
Selection conditions: ( ) Simple Search ( ) By Formula [✓ By SQL Query]

```
ENTER YOUR SQL QUERY
HERE !!!!
```

Figure 2-8 Create View window

2. Next, enter a SQL statement for the view. As described in 2.3.4, "Creating Query Views dynamically using @formulas" on page 22, you can dynamically create SQL statements based on formula language. To allow for this feature, static SQLs statements have to be enclosed by quotation marks, as shown in Example 2-1.

*Example 2-1 SQL statements must be enclosed in quotation marks*

```
"SELECT * FROM SALES.SALESDAV"
```

   You address a DB2 Access View in a database using the DB2 schema name and the DB2 Access View name: DB2SCHEMANAME.DAVNAME. In Example 2-1, the DB2 Access View is named SALES.SALESDAV. When you place a database on a server, normally the DB2 schema name is the part of the file name in front of the ".NSF". In our example, SALES.NSF becomes SALES. If the DB2 schema name is already in use, the DB2 schema name would be SALES2, SALES3, and so forth.

   **Important:** When you create the SQL statement for a Query View, you never know if the DB2 schema name at that time will be the final DB2 schema name. To ensure that you have the correct schema name, it is a best practice to use dynamic SQL statements, which we discuss in 2.3.4, "Creating Query Views dynamically using @formulas" on page 22.

3. When you finish the SQL statement, you can define the columns of the view. Do this the same way as for normal views.
4. To preview the design, click the **Notes Preview** button in the toolbar. The Refresh button in the view design will not show the view in Domino 7.

Although you might want to split your SQL statements into more than one line to make them more readable, as shown in Example 2-2, this manner of writing SQL statements will cause error messages.

*Example 2-2 This way of writing SQL causes error messages*

```
"SELECT
SALES.SALESDAV.CUSTOMERNAME
FROM
SALES.SALESDAV"
```

When we were building the ITSO Electronics application, we found that writing the SQL with lines assembled by + offered the same readability and works with Lotus Domino 7, as shown in Example 2-3. Each of the substatements has to be enclosed with quotation marks.

*Example 2-3 This way of writing the SQL statements is readable and works*

```
"SELECT "+
"SALES.SALESDAV.CUSTOMERNAME "+
"FROM "+
"SALES.SALESDAV"
```

SQL allows a simplified way to write SQL statements using aliases for the data source. The code in Example 2-3 on page 19 can also be written as in Example 2-4. Although this makes no difference in this example, it will help you to understand the more complex examples to follow.

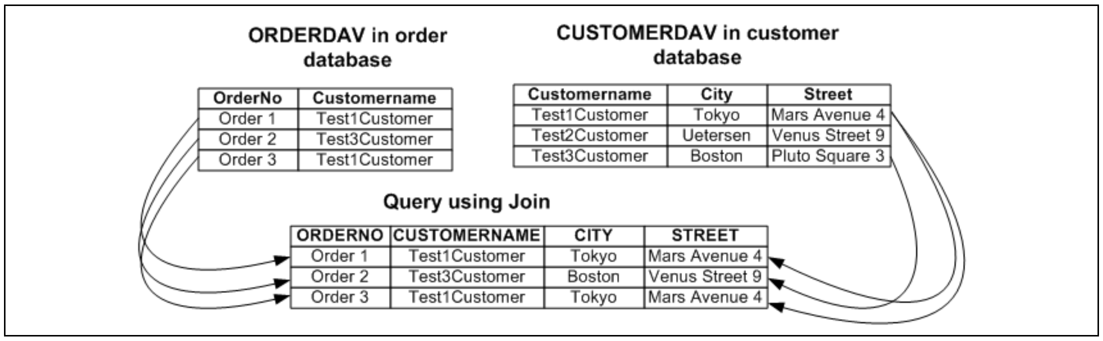
*Example 2-4 Simplified statements using aliases*

```
“SELECT “+
“S.CUSTOMERNAME “+
“FROM “+
“SALES.SALESDAV S“
```

## 2.3.2 Creating a Query View using a join

With Lotus Domino 7, you can join data from different DB2 Access Views into one Query View. It does not matter if the DB2 Access Views are in the same database or in different database. Each view entry can display connected data from different documents made available by DB2 Access Views.

Figure 2-9 shows how information from the DB2 Access Views ORDERDAV and CUSTOMERDAV is joined into a SQL ResultSet.



| OrderNo | Customername |
|---|---|
| Order 1 | Test1Customer |
| Order 2 | Test3Customer |
| Order 3 | Test1Customer |

| Customername | City | Street |
|---|---|---|
| Test1Customer | Tokyo | Mars Avenue 4 |
| Test2Customer | Uetersen | Venus Street 9 |
| Test3Customer | Boston | Pluto Square 3 |

| ORDERNO | CUSTOMERNAME | CITY | STREET |
|---|---|---|---|
| Order 1 | Test1Customer | Tokyo | Mars Avenue 4 |
| Order 2 | Test3Customer | Boston | Venus Street 9 |
| Order 3 | Test1Customer | Tokyo | Mars Avenue 4 |

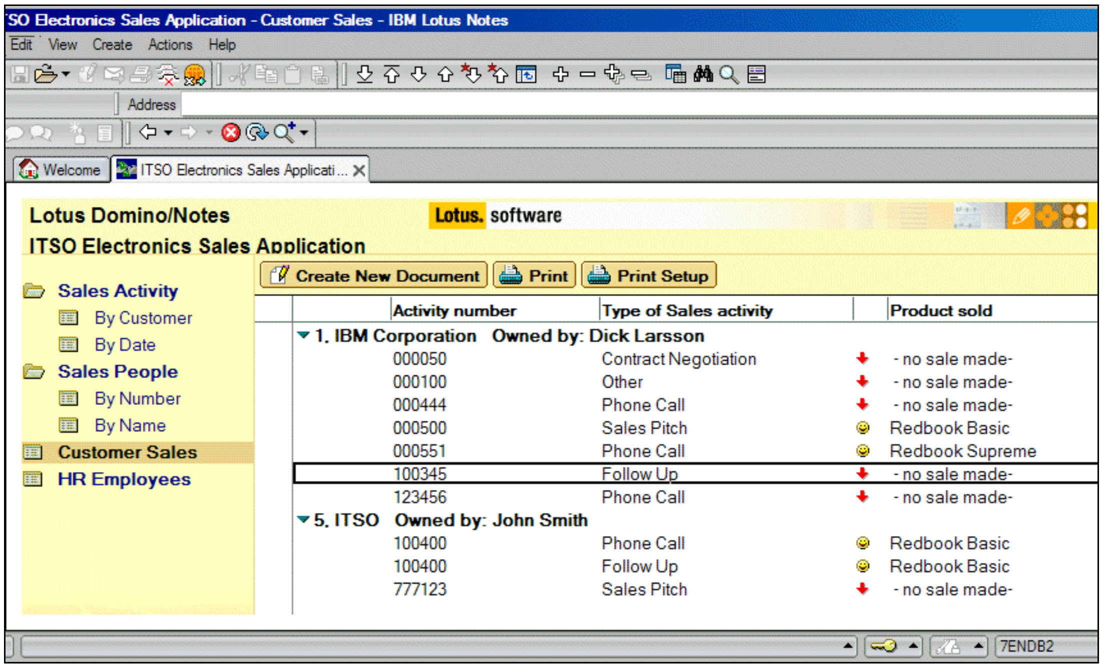
*Figure 2-9 Query Views using a join*

Example 2-5 shows the corresponding SQL statement that is used for this purpose.

*Example 2-5 SQL statement using a join to assemble a Query View*

```
"SELECT "+
"S.AType, S.Anumber,S.Product1 , C.CUSTOMERNAME, C.OWNERNAME, "+
" S.CUSTOMERNUMBER, S.MadeSale "+
"FROM "+
"SALES.SALESDAV S "+
"Left outer JOIN "+
"CUSTOMER.CUSTOMERDAV C "+
"ON "+
"S.CustomerNumber=C.CustomerNumber "+
"WHERE "+
"C.OWNERNAME<>''"
```

Figure 2-10 shows the Customer Sales view that has been created based on the SQL statement in Example 2-5 on page 20.

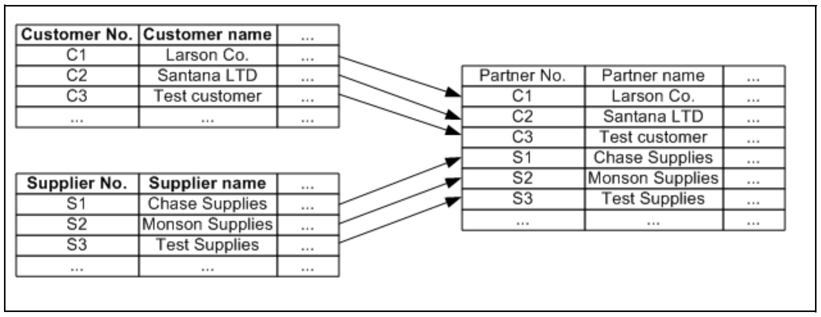


| Activity number | Type of Sales activity | | Product sold |
|---|---|---|---|
| ▼ 1. IBM Corporation Owned by: Dick Larsson | | | |
| 000050 | Contract Negotiation | ✚ | - no sale made- |
| 000100 | Other | ✚ | - no sale made- |
| 000444 | Phone Call | ✚ | - no sale made- |
| 000500 | Sales Pitch | ● | Redbook Basic |
| 000551 | Phone Call | ● | Redbook Supreme |
| 100345 | Follow Up | ✚ | - no sale made- |
| 123456 | Phone Call | ✚ | - no sale made- |
| ▼ 5. ITSO Owned by: John Smith | | | |
| 100400 | Phone Call | ● | Redbook Basic |
| 100400 | Follow Up | ● | Redbook Basic |
| 777123 | Sales Pitch | ✚ | - no sale made- |

*Figure 2-10 Query View created using a join*

## 2.3.3 Creating a Query View using a union

ITSO Electronics wants to display all external company contacts consolidated in one view. The supplier information is stored in the Products database, while the customer data is in the Customer database. In Lotus Notes and Domino Release 6, there was no way of achieving this goal without storing the data in a single database. With Lotus Domino 7, we can create a Query View using a union within our SQL statement. Figure 2-11 shows how a union assembles a Query View out of separate DB2 Access Views.



| Customer No. | Customer name | ... |
|---|---|---|
| C1 | Larson Co. | ... |
| C2 | Santana LTD | ... |
| C3 | Test customer | ... |
| ... | ... | ... |

| Supplier No. | Supplier name | ... |
|---|---|---|
| S1 | Chase Supplies | ... |
| S2 | Monson Supplies | ... |
| S3 | Test Supplies | ... |
| ... | ... | ... |

| Partner No. | Partner name | ... |
|---|---|---|
| C1 | Larson Co. | ... |
| C2 | Santana LTD | ... |
| C3 | Test customer | ... |
| S1 | Chase Supplies | ... |
| S2 | Monson Supplies | ... |
| S3 | Test Supplies | ... |
| ... | ... | ... |

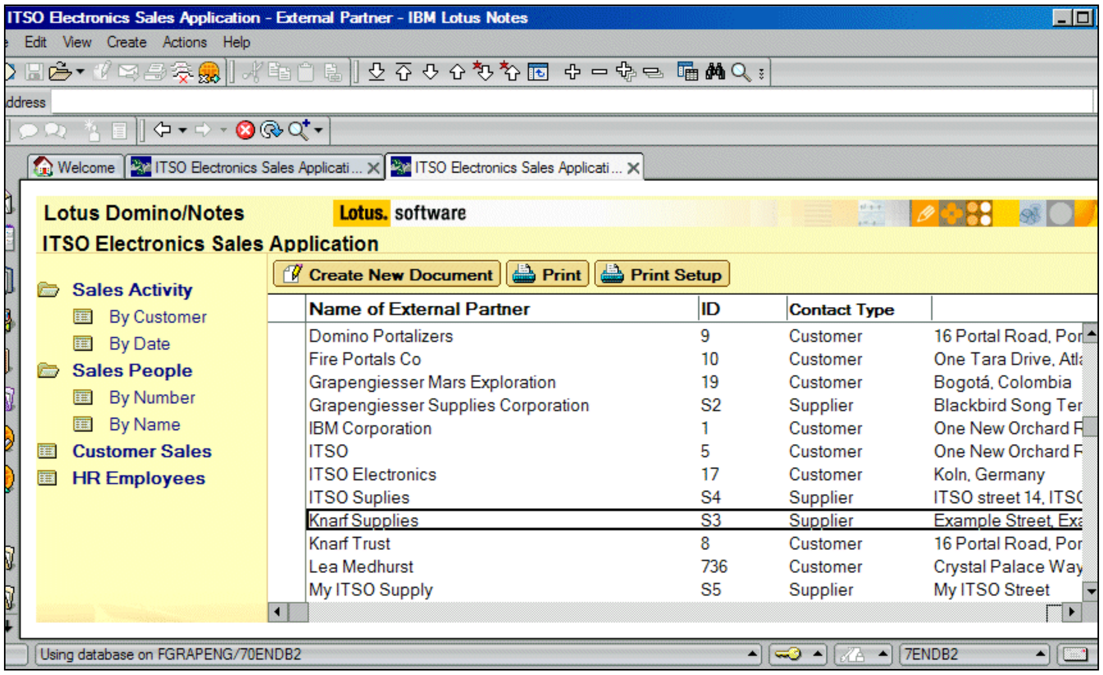
*Figure 2-11 Query Views using unions*

Example 2-6 on page 22 shows the SQL statement that is used to collect the data for ITSO Electronic's external contact view. In order for a union to work, each of the selection formulas must return the same number of columns with the same column names. Also, the corresponding columns of each results set must have matching field types.

*Example 2-6   SQL for union*

```
"SELECT " +
"C.CUSTOMERNUMBER as NO, C.CUSTOMERNAME as EXTERNALNAME, "+
"C.CUSTOMERADDRESS as ADDRESS, C.CUSTOMERDESCRIPTION as Description, "+
"C.Form as form "+
"FROM "+
"CUSTOMER.CUSTOMERDAV C " +
"WHERE "+
"C.FORM='Customer' "+
"UNION ALL "+
"SELECT "+
"S.SUPPLIERNUMBER as NO, S.SUPPLIERNAME as EXTERNALNAME, "+
"S.SUPPLIERADDRESS as ADDRESS, S.SUPPLIERDESCRIPTION as Description, "+
"S.Form as form "+
"FROM "+
"PRODUCTS.PRODUCTDAV S "  +
"WHERE "+
"FORM ='Supplier'"
```

Figure 2-12 shows a Query View displaying both the customer data and supplier data of ITSO Electronics out of different databases using the SQL code in Example 2-6.


ITSO Electronics Sales Application - External Partner - IBM Lotus Notes
Edit View Create Actions Help
Welcome
ITSO Electronics Sales Applicati...
Lotus Domino/Notes
Lotus. software
ITSO Electronics Sales Application
Create New Document
Print
Print Setup
Sales Activity
By Customer
By Date
Sales People
By Number
By Name
Customer Sales
HR Employees
Using database on FGRAPENG/70ENDB2
7ENDB2

| Name of External Partner | ID | Contact Type | |
|---|---|---|---|
| Domino Portalizers | 9 | Customer | 16 Portal Road, Por |
| Fire Portals Co | 10 | Customer | One Tara Drive, Atl |
| Grapengiesser Mars Exploration | 19 | Customer | Bogotá, Colombia |
| Grapengiesser Supplies Corporation | S2 | Supplier | Blackbird Song Ter |
| IBM Corporation | 1 | Customer | One New Orchard F |
| ITSO | 5 | Customer | One New Orchard F |
| ITSO Electronics | 17 | Customer | Koln, Germany |
| ITSO Suplies | S4 | Supplier | ITSO street 14, ITSC |
| Knarf Supplies | S3 | Supplier | Example Street, Exa |
| Knarf Trust | 8 | Customer | 16 Portal Road, Por |
| Lea Medhurst | 736 | Customer | Crystal Palace Way |
| My ITSO Supply | S5 | Supplier | My ITSO Street |

*Figure 2-12   Query View created using a union*

## 2.3.4 Creating Query Views dynamically using @formulas

So far, all of our examples use static SQL statements. In many cases, it is desirable to create SQL statements on the fly when the Query View is opened. Lotus Domino 7 enables you to calculate Query Views using @formulas.

Example 2-7 on page 23 is a simple SQL statement that collects all sales activities for the customer “ITSO” from the DB2 Access View SALESDAV. To achieve this, it uses the filter S.CUSTOMERNAME=‘ITSO’.

Example 2-7   Static SQL statement

```
“"SELECT "+
"S.CUSTOMERNAME, S.ATYPE,S.ANumber , S.MADESALE "+
"FROM "+
"SALES.SALESDAV S "+
"WHERE S.FORM='Sales Activity' AND S.CUSTOMERNAME='ITSO'"
```

Instead of this static statement, you might want to select the company name dynamically when opening the view. To achieve this goal, we add an **@Prompt** command to the code so that the end user can type the name of the company for which to look. See Example 2-8.

Example 2-8   Dynamically assemble SQL statement using formula language

```
_CUSTOMER:=@Prompt([OKCANCELEDIT];"Company selection"; "Please enter Company name you would
like to search for.";"");
"SELECT "+
"S.CUSTOMERNAME, S.ATYPE,S.ANumber , S.MADESALE "+
"FROM"+
" SALES.SALESDAV S "+
"WHERE S.FORM='Sales Activity' AND S.CUSTOMERNAME='"+_CUSTOMER+"'"
```

Each time a user opens this machine, a prompt for a customer name to be queried opens.

As already mentioned in 2.3.1, “Creating a Query View” on page 18, you cannot be sure that the SQL schema name for a database will stay the same when the database is copied or replicated onto another server. Therefore, it is advisable to avoid static references to the database schema in your SQL statements. The way around this limitation is to use the @DB2Schema function that has been introduced in Lotus Notes and Domino 7, which you may review in 4.5, “Additions to formula language” on page 92. To get the SQL DB2Schema for the current database, use **@DB2Schema** command together with the **@DBName** function. If these commands are integrated into the SQL statement from Example 2-7, we produce code similar to that shown in Example 2-9.

Example 2-9   Schema of current database is dynamically calculated

```
_Schema:= @DB2Schema(@DBname);
"SELECT "+
"S.CUSTOMERNAME, S.ATYPE,S.ANumber , S.MADESALE "+
"FROM "+
_Schema +".SALESDAV S "+
"WHERE S.FORM='Sales Activity' AND S.CUSTOMERNAME='ITSO'"
```

In other cases, you might need to write an SQL statement accessing a DB2 Access View in a different database. Assuming that the database is in the same folder as the current database, such a SQL statement based on our Example 2-7 would look like the reference in Example 2-10.

Example 2-10   Dynamic reference to database in same folder as the Lotus Domino server

```
_pathOnly:=@LeftBack(@Subset(@DbName;-1);"\\");
_path:=@If(_pathOnly="";"";_pathOnly+"\\");
_Server:=@Subset(@DbName;1);
_Schema:= @DB2Schema(_Server:@Trim(_path+"Sales.nsf"));
"SELECT "+
"S.CUSTOMERNAME, S.ATYPE,S.ANumber , S.MADESALE "+
"FROM "+
_Schema +".SALESDAV S "+
"WHERE S.FORM='Sales Activity' "
```

If when creating the SQL statement, you are not sure in which path a database you want to access is, you can use one of the following commands to get information about the database path out of configuration documents:

- **`@DBLookup`**
- **`@Environment`**
- **`@GetProfileField`**

## 2.3.5 Opening documents from Query Views

In a classic Lotus Notes view, each view entry is directly linked to a document. Therefore, you can open, copy, and delete documents from each view. This is not always the case for Query Views that are based on SQL statements. Certain conditions have to be met before you can open, copy, and delete from the Query View. The Query View row will only be linked to a document in the Notes database if you include the #NOTEID column from the DB2 Access View in your query and that #NOTEID is a valid Note ID in the currently opened Notes database.

For example, in our original sample Query View, we executed this query:

```
> SELECT * FROM SALES.SALESDAV
```

This inherently selected the #NOTEID column from the DB2 Access View and therefore created a link between the Query View rows and their associated Notes documents. The following query does not select the #NOTEID from the DB2 Access View, so rows in this Query View would not be linked to their associated Notes documents and would generate an error if you tried to manipulate them (for example, double-clicked):

```
> SELECT CUSTOMERNAME FROM SALES.SALESDAV
```

If you are selecting a limited number of columns from a DB2 Access View, you should explicitly select the #NOTEID column, for example:

```
> SELECT CUSTOMERNAME, #NOTEID FROM SALES.SALESDAV
```

If you are performing a JOIN across multiple DB2 Access Views, ensure that you explicitly choose a #NOTEID column that is a valid Note ID in the current Notes database.

The basic link between a view and a Notes document is from the #NOTEID. However, some functions, such as copy or creating a doc link, require the document's originator ID (OID). If your application needs this functionality (or you just want Query Views to interact more like regular views), enable the #OID to be exported to the DB2 Access View (using the DB2 Access View designer advanced properties) and include the #OID in your queries.

Note that if you are selecting data from multiple DB2 Access Views, you need to specify which DB2 Access View to select these columns from explicitly, for example:

```
SELECT S.AType, S.Anumber,S.Product1 , C.CUSTOMERNAME, C.OWNERNAME, C.#NOTEID, C.#OID,
S.CUSTOMERNUMBER, S.MadeSale FROM SALES.SALESDAV S Left outer JOIN CUSTOMER.CUSTOMERDAV C
ON S.CustomerNumber=C.CustomerNumber WHERE C.OWNERNAME<>''
```

## 2.3.6 Opening Query Views with Web browsers

Query Views are not available from Web browsers in the initial release of Lotus Domino 7.0. IBM will provide Query View support for Web browsers in a later release of Lotus Domino. Consult the IBM Lotus documentation site for updates on the availability of this feature at:

http://www.lotus.com/ldd/doc

## 2.3.7 Query Views and LotusScript

Provided that Query View allows you to open documents in the view, you can also access them with LotusScript (refer to 2.3.5, “Opening documents from Query Views” on page 24). In this case, the simple script in Example 2-11 will work. If you cannot open documents from the Query View, this script will throw an Object variable not set in the marked line.

*Example 2-11 Simple LotusScript looping through a view*

```
Sub Click(Source As Button)
   Dim session As New notessession
   Dim db As NotesDatabase
   Dim view As NotesView
   Dim doc As notesdocument
   Dim Item As notesitem
   Set db=session.currentdatabase
   Set view =db.Getview("<A View>")
   Set doc=view.GetFirstDocument ‘ <===============================================
   Do While Not doc Is Nothing
      Set Item =doc.GetFirstItem("<I T E M   I N   D A T A B A S E>")
      Print(item.text)
      Set doc=view.GetNextDocument(doc)
   Loop
End Sub
```

There is still a way to access data in Query Views that do not allow you to open documents with LotusScript. Access such views through the view entry class, as shown in Example 2-12.

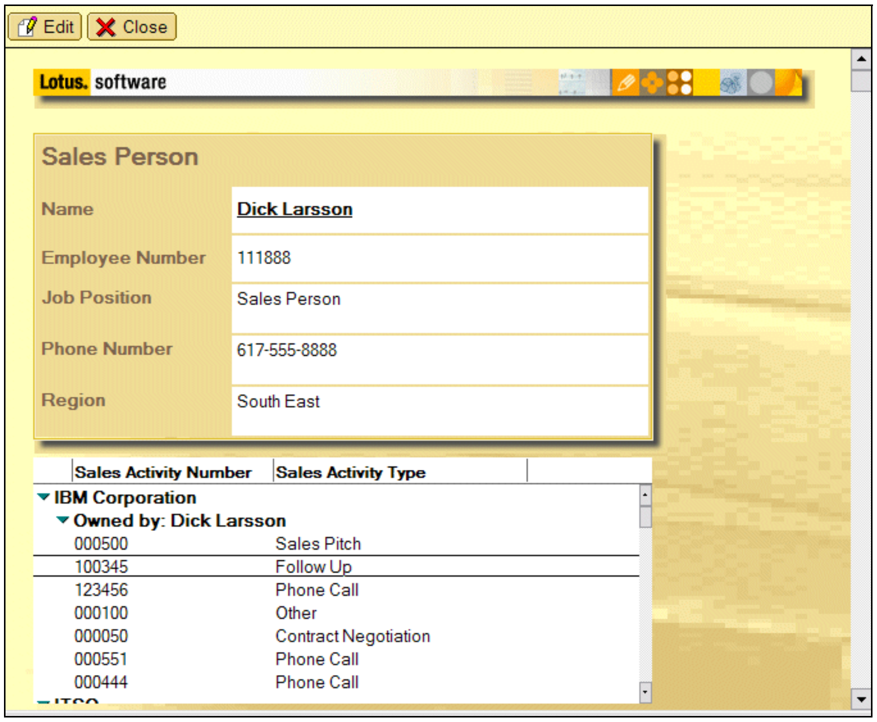
*Example 2-12 Simple script accessing view entries*

```
Sub Click(Source As Button)
   Dim session As New notessession
   Dim db As NotesDatabase
   Dim view As NotesView
   Dim doc As notesdocument
   Dim Item As notesitem
   Dim entry As NotesViewEntry
   Set db=session.currentdatabase
   Set view =db.Getview("<your viewname>")
   Set Entry = View.GetEntryByKey( "<viewEntryName>", True)
   Msgbox(entry.ColumnValues(1))
End Sub
```

## 2.3.8 Embedded Query Views in the ITSO Electronics application

As with other views, you can embed Query Views on a form, subform, page, or document. In our ITSO Electronics application, a variant of the Customer Sales Query View has been embedded inside the Sales Person form. Users accessing the Sales Person document can now easily see the information from the Customer Sales view, without closing the document, as shown in Figure 2-13 on page 26. For more information about how to create embedded views, refer to the Lotus Domino 7 Designer Help, available at:

`http://www.lotus.com/ldd/doc/domino_notes/7.0/help7_designer.nsf`



Sales Person

| | |
|---|---|
| Name | Dick Larsson |
| Employee Number | 111888 |
| Job Position | Sales Person |
| Phone Number | 617-555-8888 |
| Region | South East |

| Sales Activity Number | Sales Activity Type |
|---|---|
| IBM Corporation | |
| Owned by: Dick Larsson | |
| 000500 | Sales Pitch |
| 100345 | Follow Up |
| 123456 | Phone Call |
| 000100 | Other |
| 000050 | Contract Negotiation |
| 000551 | Phone Call |
| 000444 | Phone Call |

*Figure 2-13 Embedded Query View in Sales Person form of ITSO Electronics application*

## 2.3.9 Further considerations regarding Query Views

This section provides additional consideration about Query Views.

### Separating multivalue items

DB2 Access Views store text lists as a text string, and because Query Views read data directly from DB2, that is exactly what is returned to a Query View. So, if you have a multivalue item that you want to separate in your Query View, you must use the @Explode formula on the column that contains multiple values. In addition, to ensure that the view is set up to handle many lines of data, on the View properties → Style tab, set the Row Height and Shrink Rows to Content options to accommodate your data.

### Updating Query Views

Because Query Views can be built from a combination of Notes and federated data, how the Query View is updated depends on the nature of the data that was added or changed and how it was added.

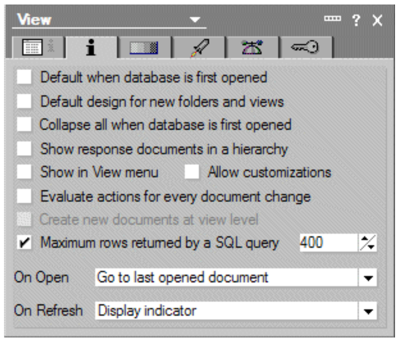
### Setting the maximum number of rows in SQL queries

By default, the maximum number of rows returned by an SQL query is 500. There are two ways to configure this value:

- The NOTES.INI parameter `DB2QueryViewRowLimit=`*`value`*

  Set the value to `0` for "unlimited" rows returned.
- On the Options tab of the View properties window, select **Maximum rows returned by a SQL query** and then specify the number of rows. See Figure 2-14 on page 27.

When both settings are in place, the smaller value is used.



- [ ] Default when database is first opened
- [ ] Default design for new folders and views
- [ ] Collapse all when database is first opened
- [ ] Show response documents in a hierarchy
- [ ] Show in View menu
- [ ] Allow customizations
- [ ] Evaluate actions for every document change
- [ ] Create new documents at view level
- [✓] Maximum rows returned by a SQL query: 400

On Open: Go to last opened document

On Refresh: Display indicator

Figure 2-14   Options tab in View properties window

## 2.4 Federated DB2 data from DB2

So far, we have discussed Query Views that show Lotus Domino data querying DB2 Access Views. However, Query Views can be built to view any data that is visible to a DB2 database, using DB2 federation, a powerful component of the DB2 and Domino integration architecture. Federated data is data stored in a database other than a Lotus Domino database (such as Oracle, Microsoft SQL, or another DB2 database) and can be made available to Lotus Domino by configuring the Lotus Domino database to support federation. You can then create Query Views that show Lotus Domino data only, federated data only, or a combination of Lotus Domino and federated data.

### 2.4.1 Enabling DB2 Notes federation

When you configure the Lotus Domino database in DB2 to support federation, which will enable you to access objects outside the current database (the database is specified in the DB2 tab of the server document in the Lotus Domino Directory), you can create what are referred to as federated objects. These DB2 federated objects enable you to not only access other DB2 tables from different DB2 databases, but other database management systems (DBMSs) as well.

In the ITSO Electronics application, we want to create a new federated Query View that accesses information from a separate DB2 database, owned by the HR department, that contains employee information. The following steps illustrate how to configure the DB2 server to allow federation:

1. Set the environment variable, DB2_ALLOW_SETAUTH_WITH_REMOTECONNECT, in order for Query Views to work. Use the following path and command:

   ```
   C:\Program Files\IBM\SQLLIB\BIN>db2set -i DB2  DB2_ALLOW_SETAUTH_WITH_REMOTECONNECT=1
   ```

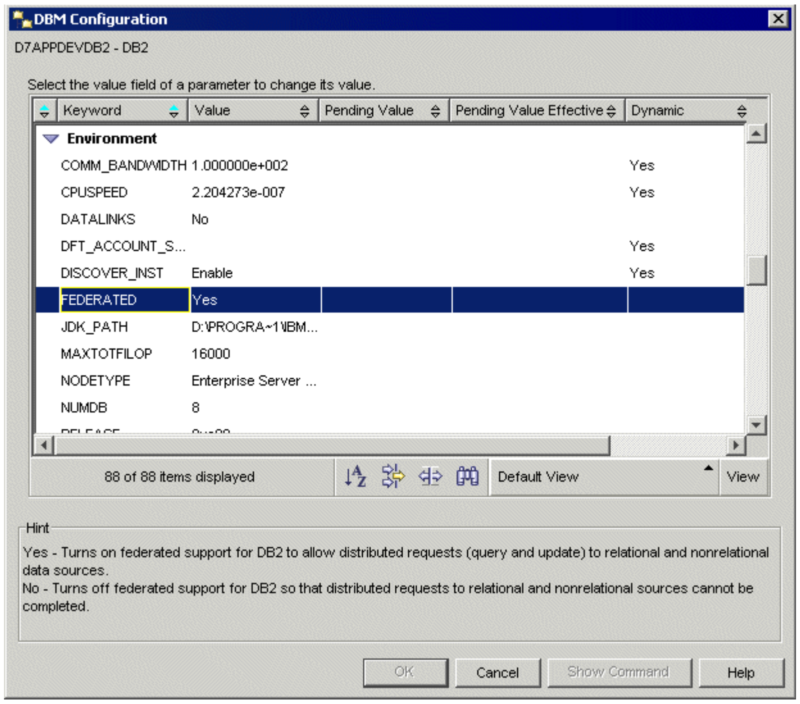
   Where `-i` specifies the name of your DB2 instance (which typically, and in our scenario, is `DB2`).

2. Restart DB2 to allow the environment variable to take effect.

   Note that you can also update the DB2 Database Manager configuration to enable federation in DB2 with the following path and command:

   ```
   C:\Program Files\IBM\SQLLIB\BIN>db2 update dbm cfg using FEDERATED YES
   ```

3. In the DB2 Control Center, connect to your Lotus Domino database. Ensure that your Lotus Domino database has federated support enabled. To verify that your Lotus Domino database is configured for DB2 federation, right-click your DB2 instance, and select **Configure Parameters**. The DBM Configuration window opens, as shown in Figure 2-15. Scroll down to the Environment section, look for the FEDERATED keyword and verify that the value is set to Yes, also as shown in Figure 2-15. If the value is No, the pending value is Yes, and you have performed the `db2set` command from step 1, you need to restart DB2 so that the value of the FEDERATED keyword will be set to Yes.



| Keyword | Value | Pending Value | Pending Value Effective | Dynamic |
|---|---|---|---|---|
| **Environment** | | | | |
| COMM_BANDWIDTH | 1.000000e+002 | | | Yes |
| CPUSPEED | 2.204273e-007 | | | Yes |
| DATALINKS | No | | | |
| DFT_ACCOUNT_S... | | | | Yes |
| DISCOVER_INST | Enable | | | Yes |
| FEDERATED | Yes | | | |
| JDK_PATH | D:\PROGRA~1\IBM... | | | |
| MAXTOTFILOP | 16000 | | | |
| NODETYPE | Enterprise Server ... | | | |
| NUMDB | 8 | | | |

Hint
Yes - Turns on federated support for DB2 to allow distributed requests (query and update) to relational and nonrelational data sources.
No - Turns off federated support for DB2 so that distributed requests to relational and nonrelational sources cannot be completed.

*Figure 2-15 DB2 Control Center: DBM Configuration*

4. To create a federation wrapper in DB2 for the external DB2 database, which in our scenario is the HR department's DB2 database, or any other third-party objects, go back to the DB2 Control Center, and under the Lotus Domino database, go to the **Federated Database Objects** folder, and click the **Create New Wrapper** action, as shown in Figure 2-16.

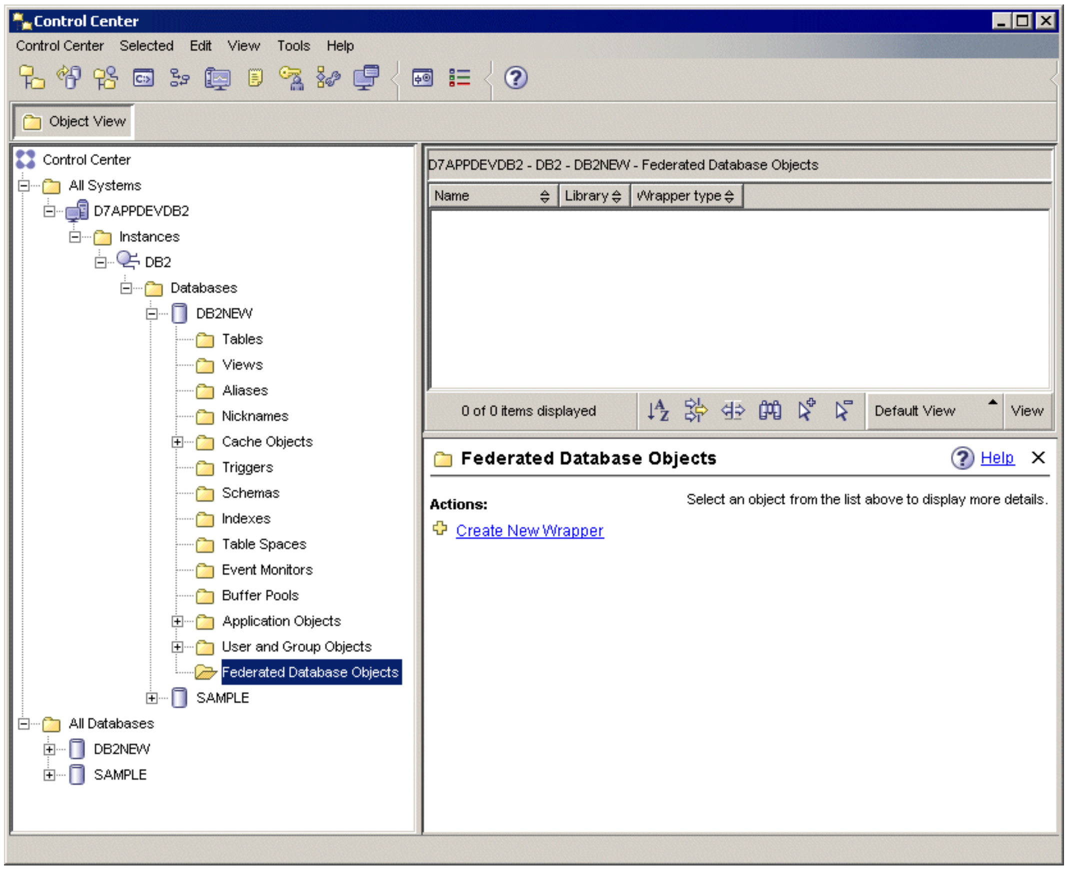
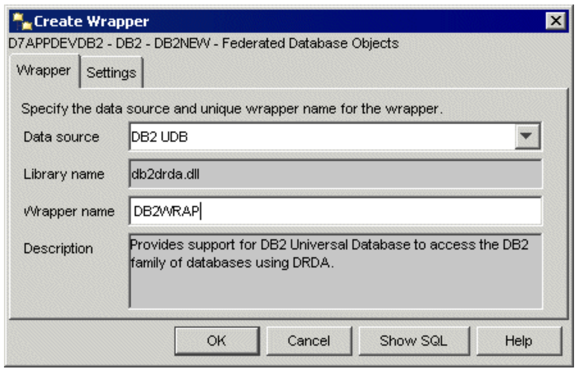


*Figure 2-16 Federated Database Objects in DB2 Control Center*

5. The Create Wrapper window opens, as shown in Figure 2-17. In the Wrapper tab, select the Data source from the drop-down list, which in our scenario is **DB2 UDB**. Specify the Wrapper name, using the value `DB2WRAP` in our setup. Click **OK**.



D7APPDEVDB2 - DB2 - DB2NEW - Federated Database Objects

Specify the data source and unique wrapper name for the wrapper.

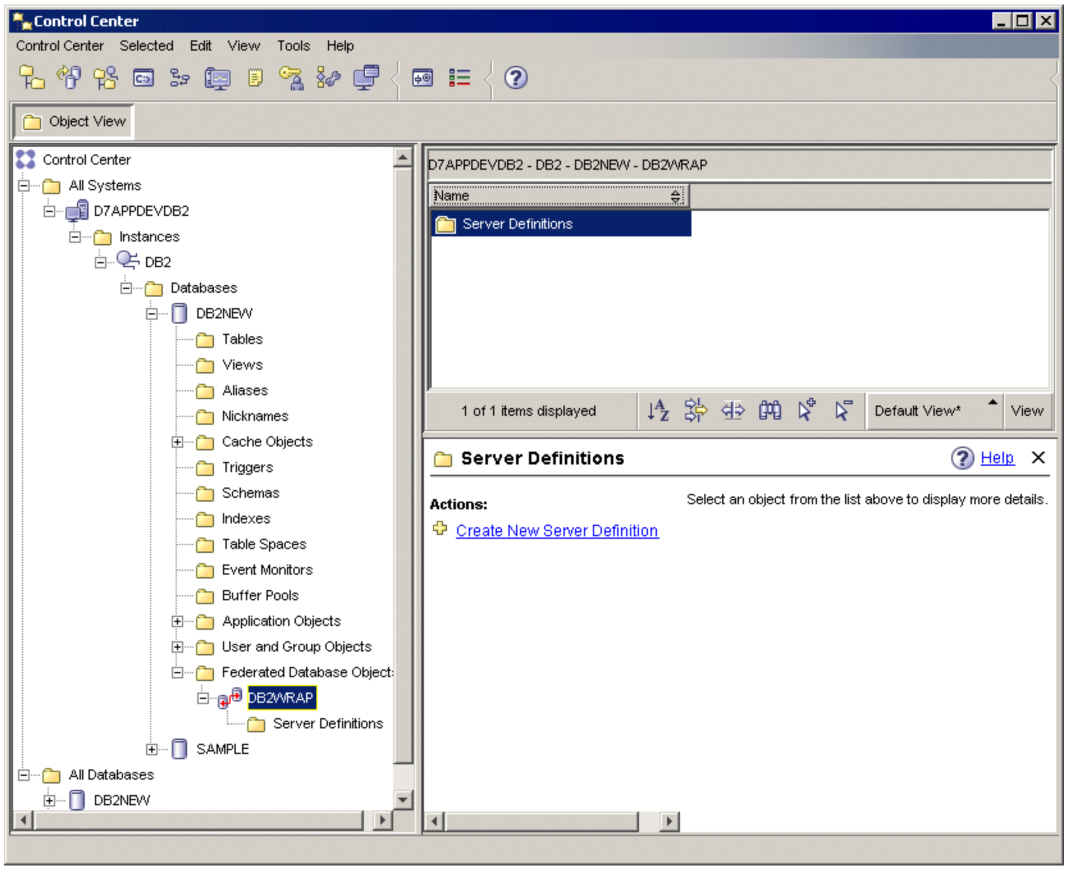
| | |
|---|---|
| Data source | DB2 UDB |
| Library name | db2drda.dll |
| Wrapper name | DB2WRAP |
| Description | Provides support for DB2 Universal Database to access the DB2 family of databases using DRDA. |

*Figure 2-17 Create Wrapper window*

**Note:** For information about wrappers and about enabling federated data support in DB2, go to the DB2 Information Center, available at:

http://publib.boulder.ibm.com/infocenter/db2help/index.jsp

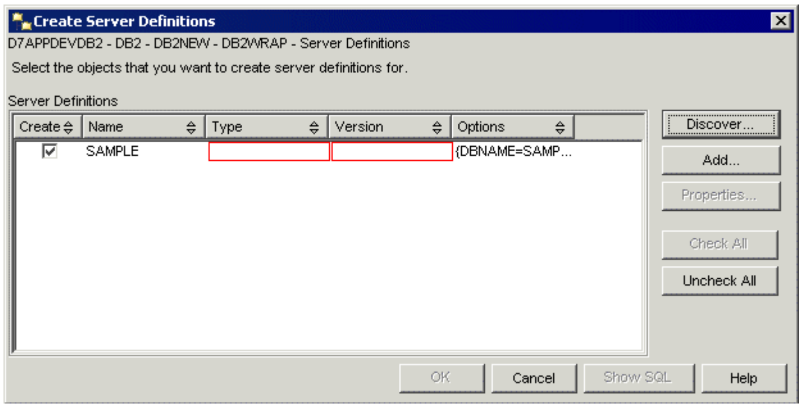
6. After the wrapper is created, it is displayed under the Federated Database Objects in the DB2 Control Center, as shown in Figure 2-18. Select the **Server Definitions** folder, and select the **Create New Server Definition** action. The Create Server Definitions window opens, as shown in Figure 2-19 on page 31.



Select an object from the list above to display more details.

*Figure 2-18 Server Definitions in DB2 Control Center*

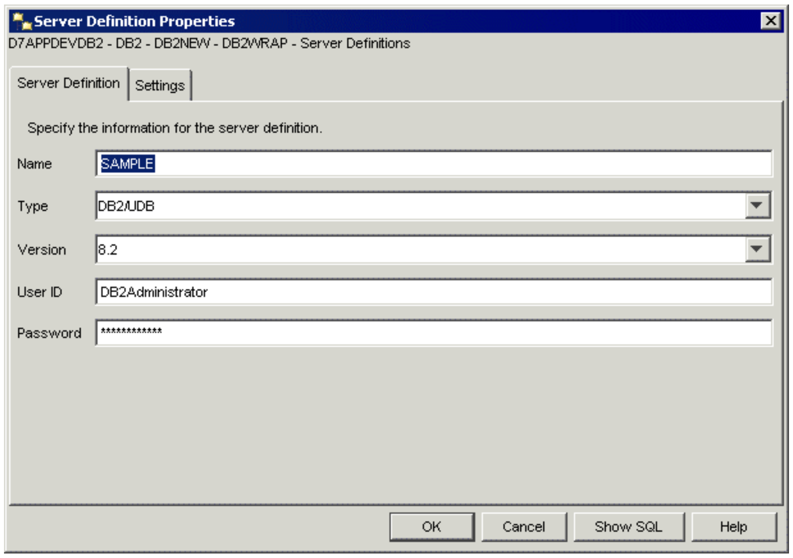
7. From the Create Server Definitions window, click **Discover**. A list of objects is displayed. Select the appropriate object, which in our scenario is the **SAMPLE** database (this is the name of the HR DB2 database), as shown in Figure 2-19.


Create Server Definitions
D7APPDEVDB2 - DB2 - DB2NEW - DB2WRAP - Server Definitions
Select the objects that you want to create server definitions for.
Server Definitions
Discover...
Add...
Properties...
Check All
Uncheck All
OK
Cancel
Show SQL
Help

| Create | Name | Type | Version | Options |
|---|---|---|---|---|
| ☑ | SAMPLE | | | {DBNAME=SAMP... |

*Figure 2-19 Create Server Definitions: Discover button*

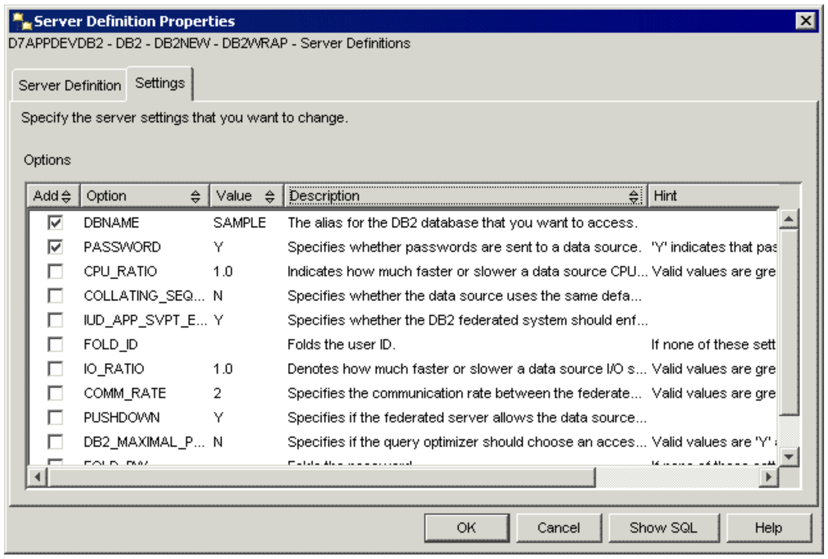
8. Click **Properties**, and the Server Definition Properties window opens, as shown in Figure 2-20. Enter the relevant values for all the fields in the Server Definition tab, which in our scenario is what is displayed in Figure 2-20. For the Type field, select **DB2/UDB** from the drop-down list, for the Version field, select **8.2** from the drop-down list, and for the User ID and Password fields, use a DB2 user (or DB2 administrator) that has access to the external DB2 database, which in our scenario is the DB2 HR database.


Server Definition Properties
D7APPDEVDB2 - DB2 - DB2NEW - DB2WRAP - Server Definitions
Server Definition
Settings
Specify the information for the server definition.
OK
Cancel
Show SQL
Help

| | |
|---|---|
| Name | SAMPLE |
| Type | DB2/UDB |
| Version | 8.2 |
| User ID | DB2Administrator |
| Password | ************ |

*Figure 2-20 Server Definition Properties window*

9. Go to the Settings tab, as shown in Figure 2-21, and enter the name of the DB2 database in the Value field for the DBNAME option. In our scenario, this is the HR DB2 database named SAMPLE. The OK button becomes available. Click **OK**.


Server Definition Properties
D7APPDEVDB2 - DB2 - DB2NEW - DB2WRAP - Server Definitions
Server Definition
Settings
OK
Cancel
Show SQL
Help

Specify the server settings that you want to change.

Options

| Add | Option | Value | Description | Hint |
|---|---|---|---|---|
| ✓ | DBNAME | SAMPLE | The alias for the DB2 database that you want to access. | |
| ✓ | PASSWORD | Y | Specifies whether passwords are sent to a data source. | 'Y' indicates that pas |
| | CPU_RATIO | 1.0 | Indicates how much faster or slower a data source CPU... | Valid values are gre |
| | COLLATING_SEQ... | N | Specifies whether the data source uses the same defa... | |
| | IUD_APP_SVPT_E... | Y | Specifies whether the DB2 federated system should enf... | |
| | FOLD_ID | | Folds the user ID. | If none of these sett |
| | IO_RATIO | 1.0 | Denotes how much faster or slower a data source I/O s... | Valid values are gre |
| | COMM_RATE | 2 | Specifies the communication rate between the federate... | Valid values are gre |
| | PUSHDOWN | Y | Specifies if the federated server allows the data source... | |
| | DB2_MAXIMAL_P... | N | Specifies if the query optimizer should choose an acces... | Valid values are 'Y' |

Figure 2-21 Server Definition Properties: Settings tab

10. We now need to add a user mapping to this federation wrapper, mapping the Lotus Domino server user to the remote connection user name and password. In our scenario, the Lotus Domino server user is DB2DOMINO and is the account that was created for the DB2 server enablement. From the DB2 Control Center, under the newly created server definition for the SAMPLE database, select the **Create New User Mapping** action, as shown in Figure 2-22.

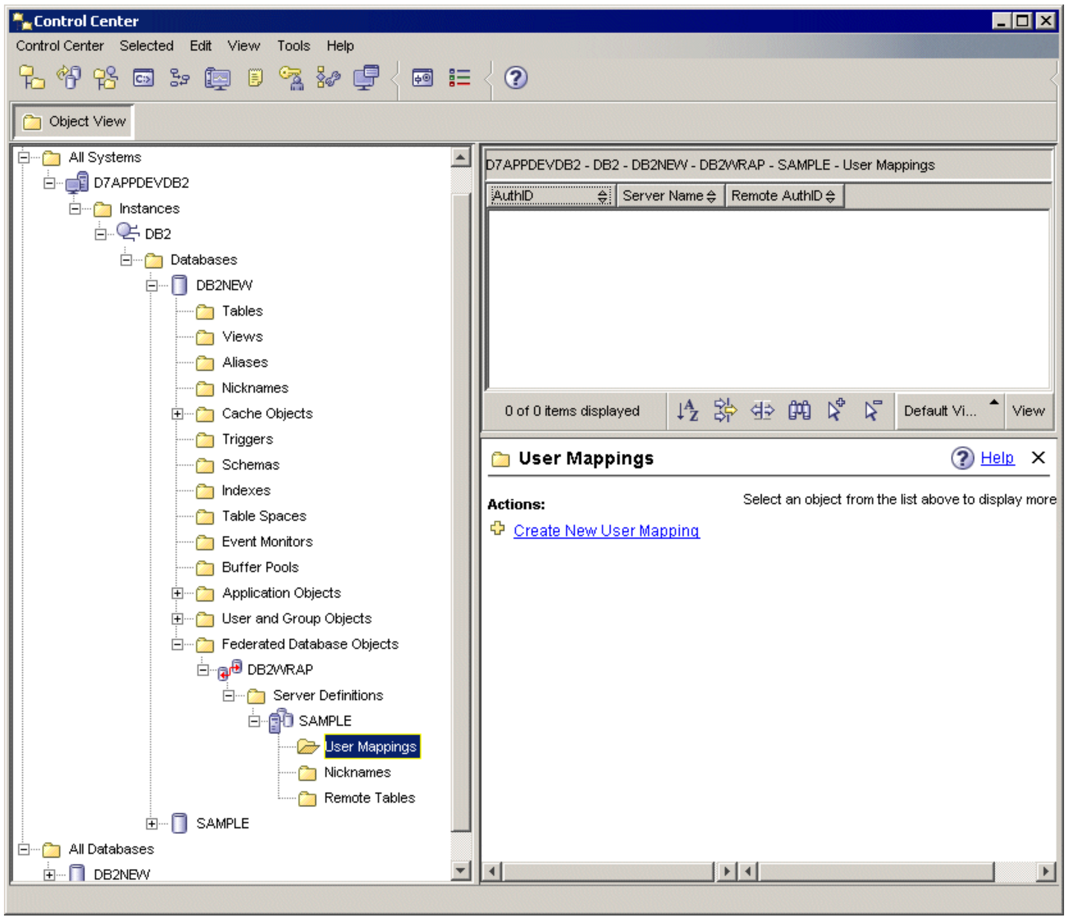
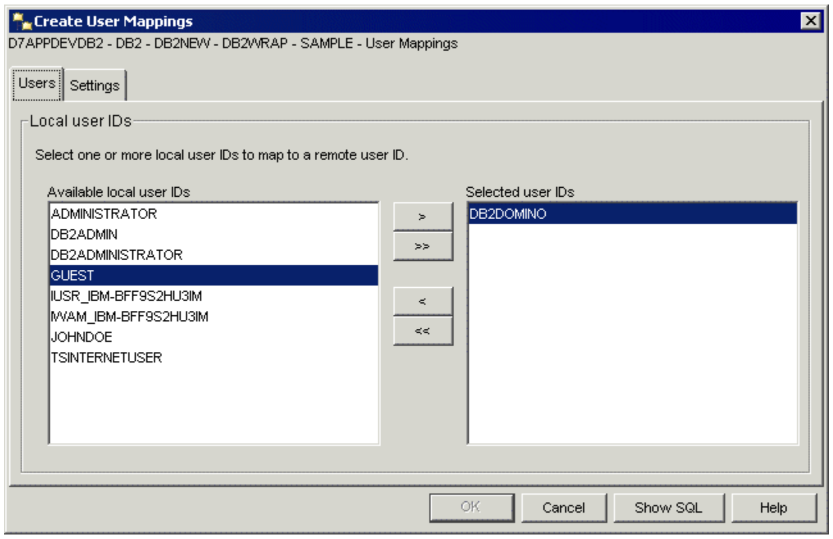


*Figure 2-22 DB2 Control Center: User Mappings*

11. The Create User Mappings window opens, as shown in Figure 2-23. From the Users tab, select one or more local user IDs to map to a remote ID. In our scenario, we select the **DB2DOMINO** user ID.



Create User Mappings

D7APPDEVDB2 - DB2 - DB2NEW - DB2WRAP - SAMPLE - User Mappings

Users | Settings

Local user IDs

Select one or more local user IDs to map to a remote user ID.

Available local user IDs:

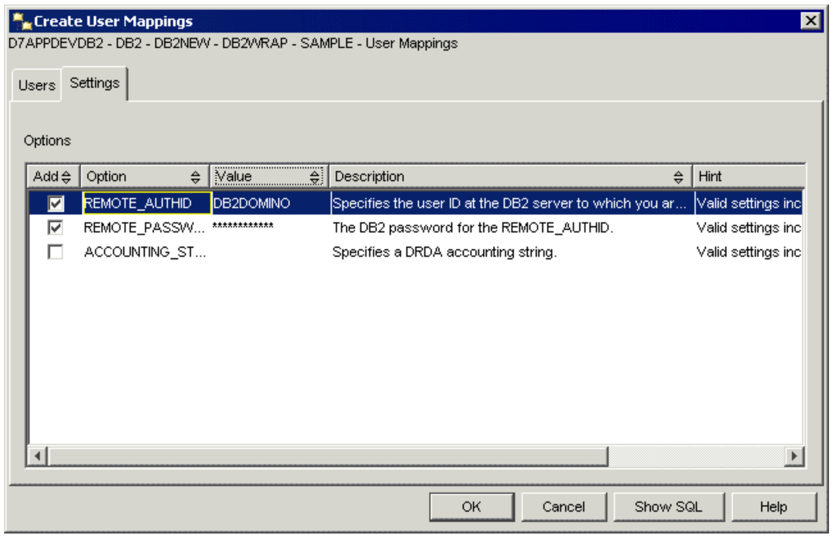
- ADMINISTRATOR
- DB2ADMIN
- DB2ADMINISTRATOR
- GUEST
- IUSR_IBM-BFF9S2HU3IM
- IWAM_IBM-BFF9S2HU3IM
- JOHNDOE
- TSINTERNETUSER

Selected user IDs:

- DB2DOMINO

*Figure 2-23 Create User Mappings window*

12. Go to the Settings tab, and enter the appropriate values for the REMOTE_AUTHID and REMOTE_PASSWORD options. In our scenario, we use DB2DOMINO and DB2DOMINO passwords, respectively, to populate the options, as shown in Figure 2-24. (In our DB2 server, we also have a DB2DOMINO user ID, and we use it in our scenario for our user mapping. However, the mapping between the user ID by which Domino connects to DB2, in our example, DB2DOMINO, to a user ID that is valid on the remote DB2 machine (REMOTE_AUTHID and REMOTE_PASSWORD options) can be distinctly named user IDs.) The OK button should then be available. Click **OK**.



Create User Mappings

D7APPDEVDB2 - DB2 - DB2NEW - DB2WRAP - SAMPLE - User Mappings

Users | Settings

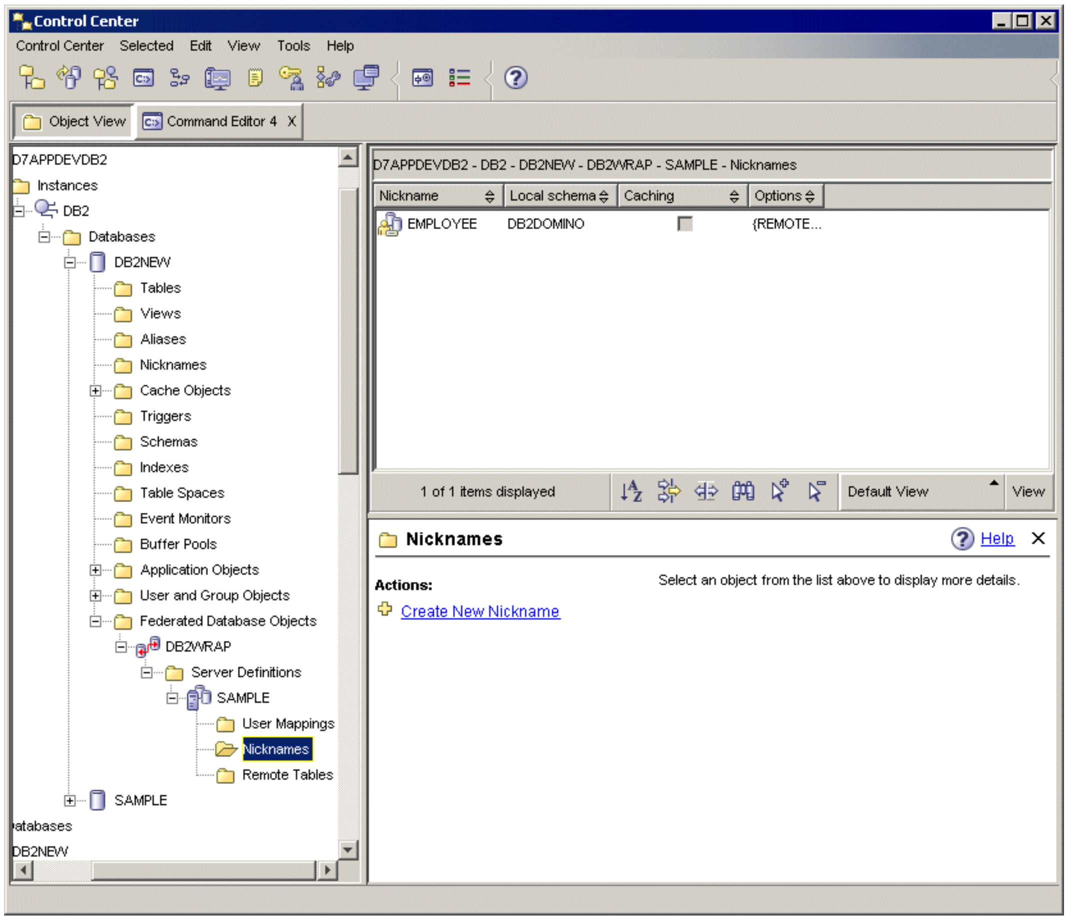
Options

| Add | Option | Value | Description | Hint |
|---|---|---|---|---|
| ☑ | REMOTE_AUTHID | DB2DOMINO | Specifies the user ID at the DB2 server to which you ar... | Valid settings inc |
| ☑ | REMOTE_PASSW... | ************ | The DB2 password for the REMOTE_AUTHID. | Valid settings inc |
| ☐ | ACCOUNTING_ST... | | Specifies a DRDA accounting string. | Valid settings inc |

*Figure 2-24 Create User Mappings: Settings tab*

**Note:** If you encounter any problems setting up the user mapping, go back to the DB2 Control Center, select the database to which you want to connect, right-click, select **Connect**, and enter the user ID and password. In our scenario application, we use the DB2DOMINO user ID and password.

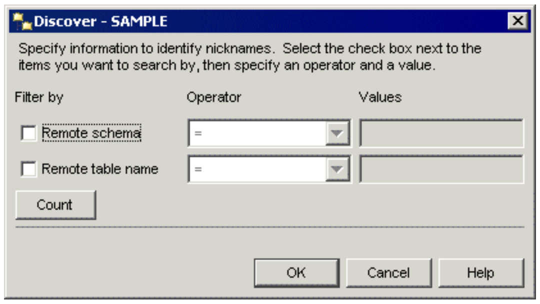
13. From the DB2 Control Center, create a nickname mapping a local object to the foreign object. The nickname is used in the Query View SQL statement. Select the **Nicknames** folder and click the **Create New Nickname** action, as shown in Figure 2-25.



| Nickname | Local schema | Caching | Options |
|---|---|---|---|
| EMPLOYEE | DB2DOMINO | | {REMOTE... |

Figure 2-25 DB2 Control Center: Create New Nickname

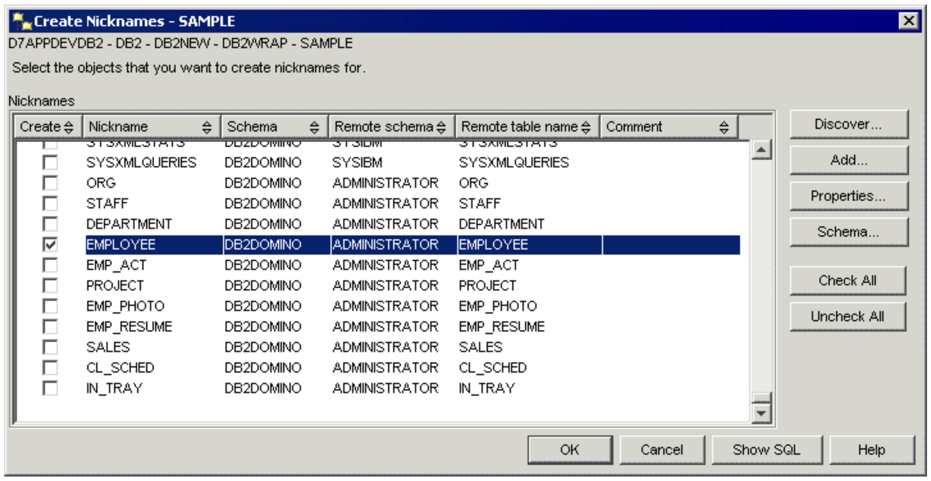
14. The **Create Nicknames** window opens. Click **Discover**, and the Discover window opens, as shown in Figure 2-26. Click **OK**.



Specify information to identify nicknames. Select the check box next to the items you want to search by, then specify an operator and a value.

Figure 2-26 Create Nicknames: Discover window

15.The Create Nicknames window is then populated with objects for which you can define nicknames. For the purposes of the ITSO Electronics application, click **Uncheck All**, and then select the **Create** option for the EMPLOYEE nickname, as shown in Figure 2-27. Click **OK**.



D7APPDEVDB2 - DB2 - DB2NEW - DB2WRAP - SAMPLE

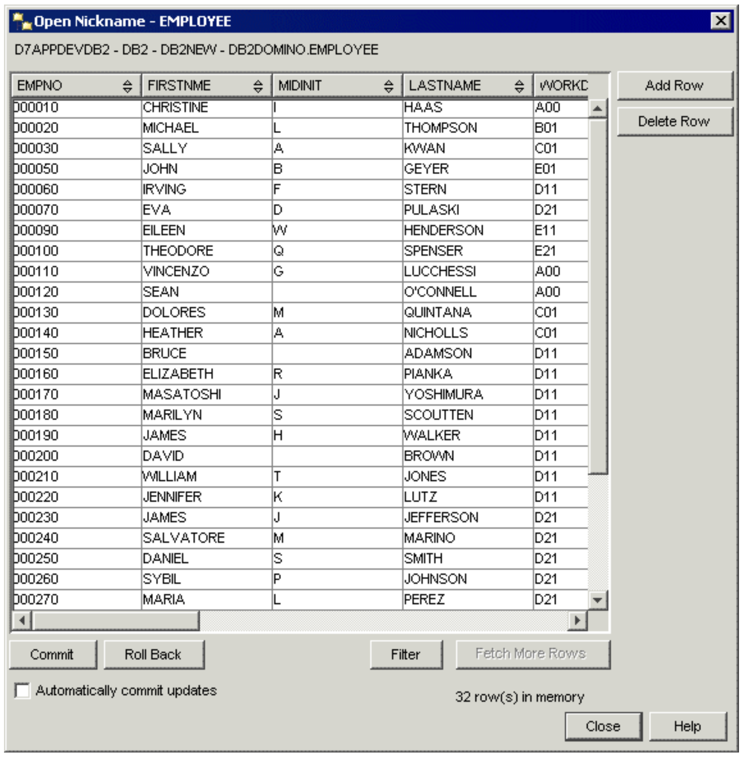
Select the objects that you want to create nicknames for.

Nicknames

| Create | Nickname | Schema | Remote schema | Remote table name | Comment |
|---|---|---|---|---|---|
| ☐ | SYSXMLSTATS | DB2DOMINO | SYSIBM | SYSXMLSTATS | |
| ☐ | SYSXMLQUERIES | DB2DOMINO | SYSIBM | SYSXMLQUERIES | |
| ☐ | ORG | DB2DOMINO | ADMINISTRATOR | ORG | |
| ☐ | STAFF | DB2DOMINO | ADMINISTRATOR | STAFF | |
| ☐ | DEPARTMENT | DB2DOMINO | ADMINISTRATOR | DEPARTMENT | |
| ☑ | EMPLOYEE | DB2DOMINO | ADMINISTRATOR | EMPLOYEE | |
| ☐ | EMP_ACT | DB2DOMINO | ADMINISTRATOR | EMP_ACT | |
| ☐ | PROJECT | DB2DOMINO | ADMINISTRATOR | PROJECT | |
| ☐ | EMP_PHOTO | DB2DOMINO | ADMINISTRATOR | EMP_PHOTO | |
| ☐ | EMP_RESUME | DB2DOMINO | ADMINISTRATOR | EMP_RESUME | |
| ☐ | SALES | DB2DOMINO | ADMINISTRATOR | SALES | |
| ☐ | CL_SCHED | DB2DOMINO | ADMINISTRATOR | CL_SCHED | |
| ☐ | IN_TRAY | DB2DOMINO | ADMINISTRATOR | IN_TRAY | |

*Figure 2-27 Create Nicknames window*

16.The EMPLOYEE nickname is then created, which is visible in the DB2 Command Center. Double-click the nickname to display the federated EMPLOYEE table, as shown in Figure 2-28.



D7APPDEVDB2 - DB2 - DB2NEW - DB2DOMINO.EMPLOYEE

| EMPNO | FIRSTNME | MIDINIT | LASTNAME | WORKD |
|---|---|---|---|---|
| 000010 | CHRISTINE | I | HAAS | A00 |
| 000020 | MICHAEL | L | THOMPSON | B01 |
| 000030 | SALLY | A | KWAN | C01 |
| 000050 | JOHN | B | GEYER | E01 |
| 000060 | IRVING | F | STERN | D11 |
| 000070 | EVA | D | PULASKI | D21 |
| 000090 | EILEEN | W | HENDERSON | E11 |
| 000100 | THEODORE | Q | SPENSER | E21 |
| 000110 | VINCENZO | G | LUCCHESSI | A00 |
| 000120 | SEAN | | O'CONNELL | A00 |
| 000130 | DOLORES | M | QUINTANA | C01 |
| 000140 | HEATHER | A | NICHOLLS | C01 |
| 000150 | BRUCE | | ADAMSON | D11 |
| 000160 | ELIZABETH | R | PIANKA | D11 |
| 000170 | MASATOSHI | J | YOSHIMURA | D11 |
| 000180 | MARILYN | S | SCOUTTEN | D11 |
| 000190 | JAMES | H | WALKER | D11 |
| 000200 | DAVID | | BROWN | D11 |
| 000210 | WILLIAM | T | JONES | D11 |
| 000220 | JENNIFER | K | LUTZ | D11 |
| 000230 | JAMES | J | JEFFERSON | D21 |
| 000240 | SALVATORE | M | MARINO | D21 |
| 000250 | DANIEL | S | SMITH | D21 |
| 000260 | SYBIL | P | JOHNSON | D21 |
| 000270 | MARIA | L | PEREZ | D21 |

☐ Automatically commit updates

32 row(s) in memory

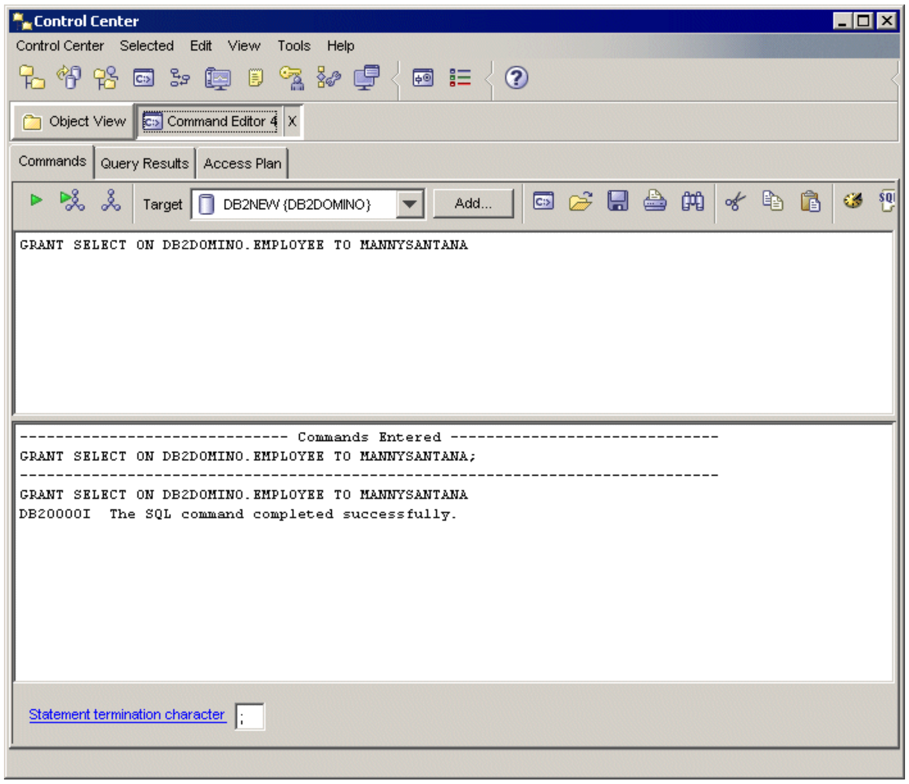
*Figure 2-28 Open Nickname: EMPLOYEE window*

17. The final step is to grant privileges to the nickname object as required for those mapped DB2 users to whom you want to give access to the federated table. In our scenario, we need to grant access to the user ID MANNYSANTANA. Complete these steps from the command prompt or from the DB2 Command Editor. Log on to your database as your Lotus Domino server user, or as another administrator. Take one of the following actions:

   – From the command prompt, use the following command to grant privileges:

     ```
     GRANT SELECT ON DB2DOMINO.EMPLOYEE TO MANNYSANTANA
     ```

   – From the DB2 Command Editor, as shown in Figure 2-29, use the same command.



```
GRANT SELECT ON DB2DOMINO.EMPLOYEE TO MANNYSANTANA
```

```
------------------------------ Commands Entered ------------------------------
GRANT SELECT ON DB2DOMINO.EMPLOYEE TO MANNYSANTANA;
------------------------------------------------------------------------------
GRANT SELECT ON DB2DOMINO.EMPLOYEE TO MANNYSANTANA
DB20000I  The SQL command completed successfully.
```

;

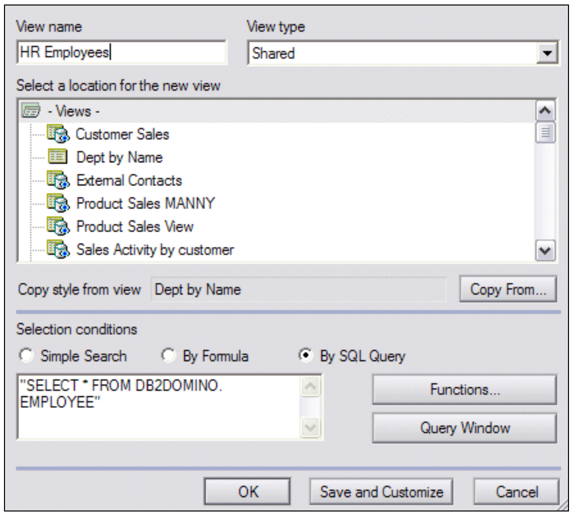
*Figure 2-29 DB2 Command Editor*

You can now create Query Views with federated data, dramatically increasing the data access capabilities of your Lotus Domino application.

## 2.4.2 Implementing DB2 Notes federation in the ITSO Electronics application

After completing the setup of our DB2 Notes federation on the DB2 server, we can access the federated data by creating federated Query Views in our ITSO Electronics application, the final stage in our implementation of the DB2 federation functionality. Follow these steps to create a federated Query View in the ITSO Electronics application:

1. Open the ITSO Electronics Sales database in Lotus Domino Designer, and select **Create** → **View**. The Create View window opens, as shown in Figure 2-30 on page 38. Enter `HR Employees` as the name of our new federated Query View, and select the **By SQL Query** radio button. Enter the SQL Query string and click **Save and Customize** to customize the view and add new columns.


View name: HR Employees

View type: Shared

Select a location for the new view

- Views -
- Customer Sales
- Dept by Name
- External Contacts
- Product Sales MANNY
- Product Sales View
- Sales Activity by customer

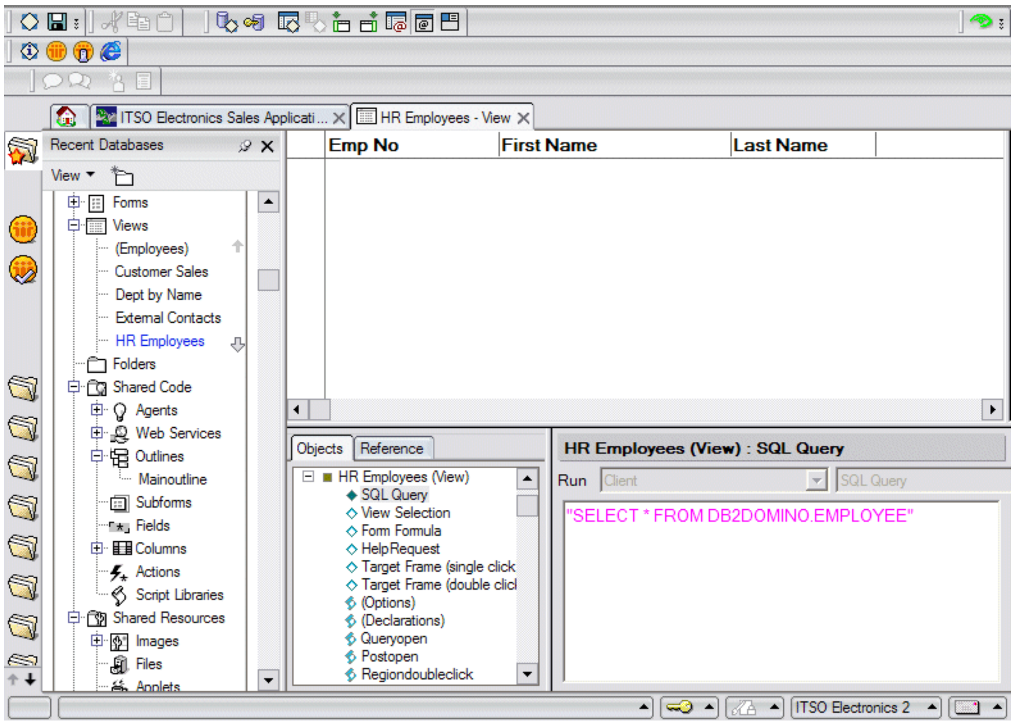
Copy style from view: Dept by Name

Selection conditions: Simple Search / By Formula / [✓ By SQL Query]

```
"SELECT * FROM DB2DOMINO.
EMPLOYEE"
```

*Figure 2-30 Create View window*

2. Add new columns to display the Employee No (number), First Name, and Last Name columns from the federated Employee table, as shown in Figure 2-31. For each column, specify the federated table column name as the column value formula.

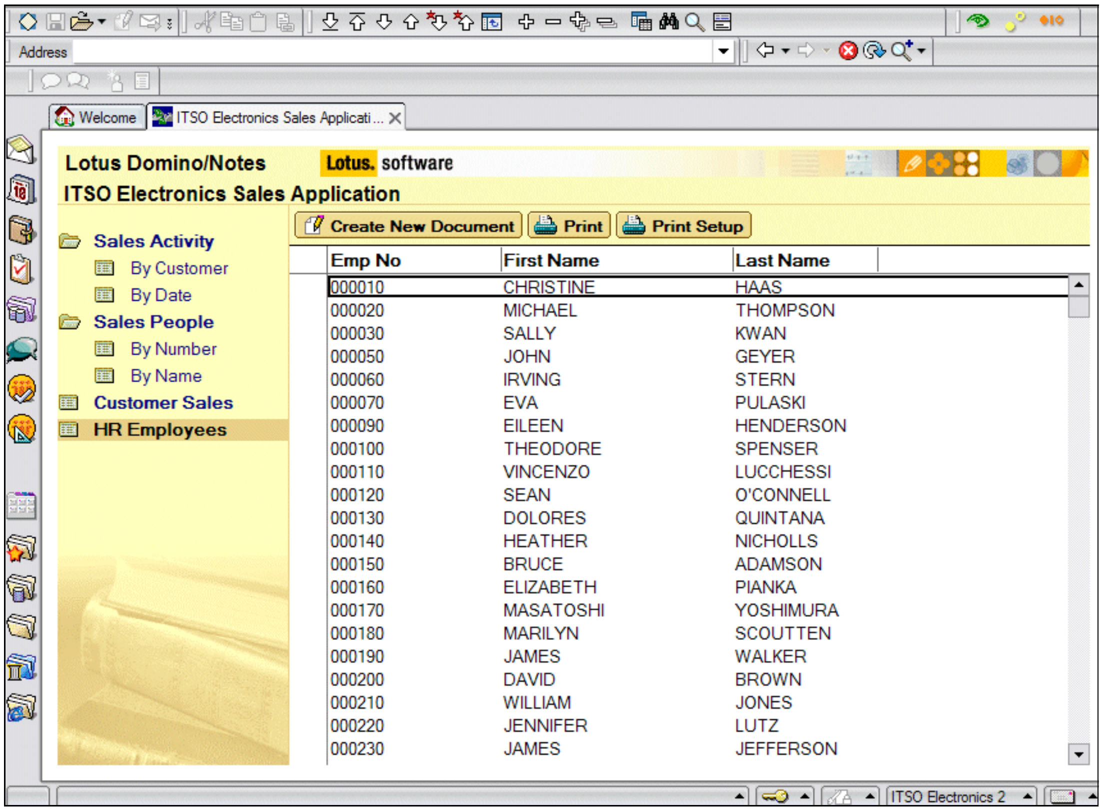


HR Employees (View) : SQL Query

```
"SELECT * FROM DB2DOMINO.EMPLOYEE"
```

*Figure 2-31 HR Employees view in Domino Designer*

3. After saving the view and adding the view to the Outline, select the HR Employees federated Query View from the Lotus Notes client to display the information from the HR Employees table, as shown in Figure 2-32 on page 39.



| Emp No | First Name | Last Name |
|---|---|---|
| 000010 | CHRISTINE | HAAS |
| 000020 | MICHAEL | THOMPSON |
| 000030 | SALLY | KWAN |
| 000050 | JOHN | GEYER |
| 000060 | IRVING | STERN |
| 000070 | EVA | PULASKI |
| 000090 | EILEEN | HENDERSON |
| 000100 | THEODORE | SPENSER |
| 000110 | VINCENZO | LUCCHESSI |
| 000120 | SEAN | O'CONNELL |
| 000130 | DOLORES | QUINTANA |
| 000140 | HEATHER | NICHOLLS |
| 000150 | BRUCE | ADAMSON |
| 000160 | ELIZABETH | PIANKA |
| 000170 | MASATOSHI | YOSHIMURA |
| 000180 | MARILYN | SCOUTTEN |
| 000190 | JAMES | WALKER |
| 000200 | DAVID | BROWN |
| 000210 | WILLIAM | JONES |
| 000220 | JENNIFER | LUTZ |
| 000230 | JAMES | JEFFERSON |

*Figure 2-32 HR Employees federated Query View*

## 2.5 Security considerations regarding DB2 and Lotus Domino

There are a number of security considerations when running DB2 with Lotus Domino:

- Even though the Lotus Domino databases are stored on the DB2 server, the access control list is still maintained, as well as the readers and authors fields.
- When you store an NSF in DB2, Lotus Domino creates and manages a set of tables to hold your NSF data, but you cannot access these tables directly. You can only access this data by defining DB2 Access Views.
- Encrypted data can be stored in a DB2-based database, but encrypted data is not accessible in DB2 Access Views.
- In Lotus Notes and Domino 7 when Notes data is accessed in SQL through a DB2 Access View, user-level security checks are always done on the user, through the Notes user name and DB2 user name mapping.
- If you access DB2 Access Views from outside Notes/Domino through SQL by default the Notes user name and DB2 name mapping is also used.
- If you do not want to use user name mapping from outside Notes, you can control access to DB2 Access Views using the DB2 GRANT mechanism by adding this setting to the NOTES.INI file on the DB2 Access server:

  ```
  Allow_Anonymous_Access_From_DB2=value
  ```

  `0` - Disables anonymous access

  `1` - Enables anonymous access through the DB2 GRANT mechanism

When this NOTES.INI setting is enabled, anonymous access to Lotus Notes is used whenever a DB2 user name is not mapped to a Lotus Notes user name.

**Note:** Although enabling anonymous access removes the name mapping requirement, Domino still controls access to the DB2 Access View. In order to use anonymous access, the Lotus Domino server must allow anonymous access and the default access level of the NSF associated with the DB2 Access View must provide sufficient rights to perform the requested SQL operation, for example, read permission for SELECTs, author permission for INSERTs, and so forth.

- If you try to look at a DB2 Access View in the DB2 Control Center and the DB2 account you are using does not allow access to the DB2 Access View in Lotus Domino, you will get the following error message:

```
“com.ibm.db.DataException: A database manager error occurred.: [IBM][CLI Driver][DB2/NT]
SQL0443N  Routine "ISREADER3" (specific name "") has returned an error SQLSTATE with
diagnostic text "14512 : No matching Notes user found for DB2 user and anony".
SQLSTATE=04004
```

Further information about the topic security and DB2 integration is available in the DB2 Access View security topic in Lotus Domino 7 Designer Help, at:

http://www.lotus.com/ldd/doc/domino_notes/7.0/help7_designer.nsf

Here, we only discuss security aspects of the DB2 and Domino integration. For a broad overview of security and Lotus Domino, refer to the forthcoming Redpaper *Security Considerations in Notes and Domino 7*, REDP-4104.

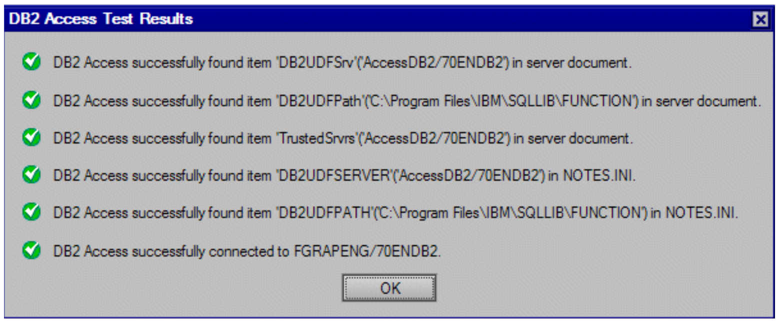
## 2.6 Troubleshooting DB2 and Lotus Domino

In this section, we provide some guidance for troubleshooting problems with DB2 integration. In addition, refer to the DB2 Integration and Troubleshooting topic in Lotus Domino 7 Designer Help.

If you encounter problems with your Domino DB2 integration, the first thing to do is to test the configuration of the DB2 Access server. The DB2 Access Test tool tests all the DB2 Access server's field parameters from the Server document and all DB2 Access server settings from the NOTES.INI file. If all fields and settings are correct, it tests the connection between the DB2 Access server and the selected Lotus Domino server, verifies whether the DB2 functions and properties exist, determines whether the DB2 Access server Connection document is valid, and attempts to open the Lotus Domino Directory on the DB2 Access server. If the DB2 Access Test tool locates any problems, the information is returned to the Lotus Domino server console or to the Lotus Domino Administrator client. To use the DB2 Access Test tool:

1. From the Lotus Domino Administrator, go to the Configuration tab.
2. Click **Connections**. Select the DB2 Access Server Connection document.
3. From the Tools panel, click **DB2 Server** → **Test DB2 Access**.

Figure 2-33 on page 41 shows the output of the DB2 Access Test tool.



```
DB2 Access successfully found item 'DB2UDFSrv'('AccessDB2/70ENDB2') in server document.
DB2 Access successfully found item 'DB2UDFPath'('C:\Program Files\IBM\SQLLIB\FUNCTION') in server document.
DB2 Access successfully found item 'TrustedSrvrs'('AccessDB2/70ENDB2') in server document.
DB2 Access successfully found item 'DB2UDFSERVER'('AccessDB2/70ENDB2') in NOTES.INI.
DB2 Access successfully found item 'DB2UDFPATH'('C:\Program Files\IBM\SQLLIB\FUNCTION') in NOTES.INI.
DB2 Access successfully connected to FGRAPENG/70ENDB2.
```

*Figure 2-33 Output of DB2 Access Test tool*

## 2.6.1 ACLs needed for DB2 Access Views

When using DB2 Access Views, if users encounter problems while attempting to access your DB2-enabled Lotus Notes database, make sure that the ACL entry for each user is identical to the first entry in the “User Name” field in that user's Person document in the Lotus Domino Directory on the Domino server.

## 2.6.2 Errors while opening a Query View

Generally, when you receive an error message when opening a Query View, it is a good idea to have a look at the server console of the Lotus Domino server. The error message on the server console often directly points to the root cause of the problem.

The following list presents common error messages and some information around them:

- DB2-Authorization Violation.

  Even though the error message suggests otherwise, this problem often has nothing to do with a security problem. Looking at the server console will give you more information:

  – Failed when DB2 was collecting column info. DB2-Authorization violation.: 42S02 - [IBM][CLI Driver][DB2/NT] SQL0204N "SALES.SALESDAV" is an undefined name. SQLSTATE=42704 - *LOCAL.DB2.051103024144

    There are three possible reasons for this error message:

    • The DB2 schema Sales in this example does not exist.

    • The DB2 access SALESDAV in this example does not exist.

    • The DB2 Access View has not been populated.

  – Failed when DB2 was collecting column info. DB2-Authorization violation.: 42S22 - [IBM][CLI Driver][DB2/NT] SQL0206N "C.CUSTOMERNAM" is not valid in the context where it is used. SQLSTATE=42703 - *LOCAL.DB2.051103024144

    One reason for this error is that the column that is called does not exist in the DB2 Access View, in this example, CUSTOMERNAM.

- DB2-Dynamic SQL Error.

  This error message points to a syntax error in the SQL statement. Look at the server console and check the SQL statement:

  Failed when DB2 was collecting column info. DB2-Dynamic SQL error.: 42601 - [IBM][CLI Driver][DB2/NT] SQL0104N An unexpected token "FROMSALES2" was found following "SELECT * ". Expected tokens may include: "<from>". SQLSTATE=42601 - G921557D.GD04.0088C6170623

In this example, there was a missing space between FROM and SALES2 in the command.

- Opening View: The given query cannot generate a result set and therefore has not been executed.

  For security reasons, SQL statements in Query Views have been limited to statements that return result sets. Update or Insert commands do not work when used as a selection formula. For example, if the first command is a SELECT, a typing error at the start of the SQL statement, such as writing SELCT instead of SELECT, can cause this error.

- This function is not implemented on this version of the server.

  You are probably trying to open a Query View on a local replica of the server or on a system that is not DB2 enabled.

- Failed when DB2 was fetching a note. DB2-External function exception.: 38503 - [IBM][CLI Driver][DB2/NT] SQL0430N User defined function "DOMINO.ISREADER3" (specific name "SQL051024022301200") has abnormally terminated. SQLSTATE=38503 - *LOCAL.DB2.051028175704

  One reason for this problem to occur is an earlier version of Lotus Notes and Domino on the system. Removing this earlier version, or renaming the folder, solves the problem in these cases.

**Tip:** Simplify your code or application design step-for-step until the problem does not occur anymore. Very often this shows the root cause of the problem. In many cases, a workaround for the problem becomes obvious.

### 2.6.3 Data to collect that might be helpful to find the root cause of problems

It is often helpful to collect the troubleshooting information. This will help you, your colleagues, or IBM personnel to find clues for the causes of your problem:

- Exact version information about the software components you are using:
  - Operating system
  - DB2 server
  - Runtime client
  - Lotus Notes and Domino
  - DB2 Access server
- Patch level information and information about localizations used.
- Screen captures of all error message.
- In case of a crash of one of the following components, a NSD-file with diagnostic information will be created if any of the following components are failing:
  - Lotus Notes client
  - Lotus Domino server
  - DB2 Access server
- All available log information.
- The database design.
- A step-by-step description of what you where doing before the problem occurs.
- NOTES.INI of Lotus Domino server and DB2 Access server.

3

# Enhancing the ITSO Electronics application with Web services

With the addition of Web services, IBM Lotus Domino Designer 7 continues its legacy of supporting new standards to extend its rapid application development model. Web services provide the ability to turn Lotus Domino applications into Web services hosted on a Lotus Domino server, enabling increased data access and manipulation across a variety of platforms. Because they are built on open standards, Web services are self-contained, modular applications that are able to work together without relying on custom-coded connections. Because Web services allow different applications and IT systems to interact with each other, service components can be combined and recombined on the fly, enabling companies to respond rapidly to changing business and customer needs.

In this chapter, we examine how the ITSO Electronics application is enhanced by the implementation of Web services by focusing on the following areas:

- Service-oriented architecture: How Web services fits into the SOA scheme
- Web services and Web Services Description Language (WSDL)
- Business reasons for implementing Web services
- Web service element: Adding simple and complex Web service elements to the ITSO Electronics application
- Exporting and importing a WSDL document
- Exception and error handling in Web services
- Web services security
- Consuming Web services using Java and LS2J/LotusScript

## 3.1 Service-oriented architecture and Web services

Service-oriented architecture (SOA) is a component model that interrelates an application's different functional units, called services, through well-defined interfaces and contracts between these services. The interface is defined in a neutral manner that should be independent of the hardware platform, the operating system, and the programming language in which the service is implemented. This allows services, built on a variety of such systems, to interact with each other in a uniform and universal manner. Because they epitomize the philosophy of neutral, platform-independent application development, Web services are an integral component of the SOA application architecture.

This feature of having a neutral interface definition that is not strongly tied to a particular implementation is known as loose coupling between services. The benefit of a loosely-coupled system is its agility and ability to survive evolutionary changes in the structure and implementation of the internals of each service that make up the whole application. Tight-coupling, however, means that the interfaces between the different components of an application are tightly interrelated in function and form, thus making them brittle when any form of change is required to parts of or the whole application.

The need for loosely-coupled systems rose from the need for business applications to become more agile based on the needs of the business to adapt to its changing environment, such as changing policies, business strengths, business focus, partnerships, industry standing, and other business-related factors that influence the very nature of the business. You can refer to a business that can act flexibly in relation to its environment an On Demand Business, where change occurs in how things are done or work as necessary in an on demand manner.

Service-oriented architecture is not new, but is an alternative model to the more traditionally tightly-coupled, object-oriented models that have emerged in the past decades. While SOA-based systems do not exclude the fact that individual services can themselves be built with object-oriented designs, the overall design is service-oriented. Because it allows for objects within the system, SOA is object-based, but it is not, as a whole, object-oriented. The difference lies in the interfaces themselves. A classic example of a proto-SOA system that has been around for a while is the Common Object Request Broker Architecture (CORBA), which defines similar concepts to SOA.

However, the SOA of today is different in that it relies on a more recent advance based on the Extensible Markup Language (XML). By describing interfaces in an XML-based language called Web Services Definition Language (WSDL), services have moved to a more dynamic and flexible interface system than the earlier Interface Definition Language (IDL) found in CORBA.

Aside from Web services, there are other ways implementing SOA, such as CORBA and message-oriented middleware systems such as IBM WebSphere® MQ. But to become an architecture model, you need more than just a service description. You need to define how the overall application performs its workflow between services. More so, you need to find the transformation point between the operations of the business versus the operations of the software used in the business. Therefore, an SOA should be able to relate the commercial processes of a business like ITSO Electronics to their technical processes and map the workflow relationships between the two. For example, the act of paying a supplier is a business process, while updating your electronic parts database to include the newly supplied shipment is a technical one. Thus, workflow also plays a significant role in the design of SOA.

## 3.2 Web service element and WSDL

A Web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards. Lotus Domino Designer 7 includes a Web service design element and built-in support for Web Services Description Language (WSDL). WSDL is a standard specification for describing networked, XML-based services. It provides a simple way for service providers to describe the basic format of requests to their systems regardless of the underlying runtime implementation.

Lotus Domino 7 restricts the Web services implementation as follows:

- Only provider entities are supported natively.
- Binding must be Simple Object Access Protocol (SOAP) using HTTP POST protocols.

The Web service element has the following characteristics:

- The compiled Web service, like a Web agent, is a stand-alone program in a Lotus Domino database. To be used, the Web service must be on, or be replicated to, a server that is enabled for Web access. It can then be accessed with the following Lotus Domino URL commands:
  - **`?OpenWebService`**: Using the HTTP POST protocol, invokes the Web service. If the HTTP GET protocol is used, the **`?OpenWebService`** command returns some information about the service.
  - **`?WSDL`**: Queries the Web service for its WSDL document.
- A Web service can be tested through an HTTP session in a Lotus Notes or Lotus Domino Designer preview.
- A Web service has the same security capabilities as an agent.
- You can import an existing WSDL document to generate a skeleton Web service. The skeleton code corresponds to the Web service definition. You then add implementation code.
- Alternatively, you can start a Web service from scratch, writing your own Java or LotusScript code to create the Web service definition.

### 3.2.1 WSDL document

A WSDL document defines a Web service and incorporates the following elements:

- Service

  The Web service as a whole. In a WSDL document, the service is described by a <definitions> element at the root. Children are <types>, <message>, <portType>, <binding>, and <service>. The <service> element has a “name” attribute that names the service.

- Types

  The data types used by the service. In a WSDL document, the data types are described in a <types> element containing one or more <xsd:schema> elements. For descriptions of the data types, see:

  `http://www.w3.org/TR/xmlschema-2`

- Message

  An abstract definition of data being communicated to or from the service. There can be any number of messages. In a WSDL document, each message is described by a <message> element. Each <message> element contains one or more <part> elements to define the data by name and type.
- PortType

  An abstract set of operations supported by the service. In a WSDL document, a port type is described by a <portType> element. Each <portType> element contains one or more <operation> elements.
- Operation

  An abstract description of an action supported by the service. In a WSDL document, an operation is described by an <operation> element. Each <operation> element contains <input> and <output> elements which specify associated messages. Optional <fault> elements may also be specified.
- Binding

  A concrete protocol and data format specification for a port type. In a WSDL document, a binding is described by a <binding> element. A <binding> element has a “type” attribute that specifies the name of the port type. For SOAP encoding, a <binding> element contains a <soap:binding> element with “style” (“rpc” or “document”) and “transport” (“http://schemas.xmlsoap.org/soap/http”) attributes. A <binding> element contains <operation> elements that describe the data format for each operation.
- Port

  A single endpoint defined as a combination of a binding and a network address. In a WSDL document, a port is defined by a <port> element under the <service> element. The <port> element has a "binding" attribute to name the binding and an "address location" attribute to name the network endpoint to be associated with the binding.

### 3.2.2 Simple Object Access Protocol

The Lotus Domino 7 Web services element provides a hosting environment based on Simple Object Access Protocol (SOAP) 1.1. SOAP defines XML-based information used for exchanging structured and typed information between peers in a decentralized, distributed environment. A SOAP message is fundamentally a one-way transmission from a SOAP sender to a SOAP receiver, but SOAP messages are expected to be combined by applications to implement more complex interaction patterns.

From the draft World Wide Web Consortium (W3C) specification:

> “SOAP is a lightweight protocol for exchanging structured information in a decentralized, distributed environment. It is an XML based protocol that consists of three parts: an envelope that defines a framework for describing what is in a message and how to process it, a set of encoding rules for expressing instances of application-defined data types, and a convention for representing remote procedure calls and responses.”[1]

### 3.2.3 Imported WSDL

In Lotus Domino Designer 7, the application developer can import an existing WSDL file into the application and Domino Designer will then generate LotusScript or Java classes. However, application developers should be aware that not all existing WSDL files can be

[1] From `http://www.w3.org/Submission/2000/05/`

interpreted by Lotus Domino Designer 7, and some are rejected, depending on the XML elements used in the file.

## 3.3 Business reasons for implementing Web services

Starting with Domino Release 5,applications could be written outside of the Lotus Domino server and could access Lotus Domino data directly. This was done using the COM bindings and the CORBA bindings to the back-end classes. However, the problem of data validation and data consistency might not have been applied consistently or correctly by these external applications. Web services provide a very good compromise between direct data access and the capability of performing network operations. They provide a way for application functions and data to be exposed to the network. Developers of Java, Microsoft .NET, and other applications can access Lotus Domino data simply by consuming Web services.

The advantage of a Lotus Domino application designed around Web services can best be exemplified in the case of the ITSO Electronics application that provides customers with current product information. For example, if the ITSO Electronics application promises users access to their product information, it needs to allow external applications access to the customer database, regardless of whether the application was created using Java, J2EE, or .NET. Because the customer's application sends a request for information to the Web service, it is completely platform and operating system agnostic. The Web service element receives the data request as an XML message. Therefore, if a user enters a query in the customer's application form to obtain their product information, the Web service receives the request as a SOAP message in XML format and sends a response to the customer in the same manner. Using extended support, the ITSO Electronics application can expand its functionality to obtain pricing information from a multitude of manufacturers. Using Java, the Web service element can send data requests to a multitude of URL addresses asking for pricing data using the XML protocol. The requests are all sent as SOAP messages and the responses received in the same format, with the ITSO Electronics application and the user having no knowledge of the database type from which it receives data, the operating system, or platform. Users see only that they has access to a seemingly limitless store of data, presented with impressive efficiency.

## 3.4 Adding Web services to the ITSO Electronics application

Using the Web service design element and WSDL support, you can rapidly create or extend Java or LotusScript code with Web services, thereby exposing thousands of Lotus Domino applications to external systems, such as Java 2 Platform, Enterprise Edition (J2EE) applications and Microsoft .NET applications. In this section, we demonstrate how the ITSO Electronics application is enhanced by implementing Web services in the Lotus Domino application development environment.

**Note:** Web services can be added to any Lotus Notes and Domino databases; however, for the purposes of this Redpaper, all Web services are added to separate databases in the ITSO Electronics application called webservices.nsf and wsconsumer.nsf.

### 3.4.1 Business requirements for adding Web services to ITSO Electronics

Our main business requirement for implementing Web services in the ITSO Electronics application was the necessity of allowing customers to view current product information. Web services met this need perfectly. In the future, ITSO Electronics plans to use Web services to allow customers to access invoice information independent of their platform, operating system, or interface.

### 3.4.2 User requirements

The primary user of Web services in the ITSO Electronics application in our example is an ITSO Electronics customer. The application needed to allow customers to access the current product information through a programmatic mechanism and through a user interface that was intuitive and easy-to-use.

## 3.5 Creating a simple Web service in Lotus Domino Designer 7

Let us take a look at how easy it is to create a Web service using IBM Lotus Domino Designer 7.

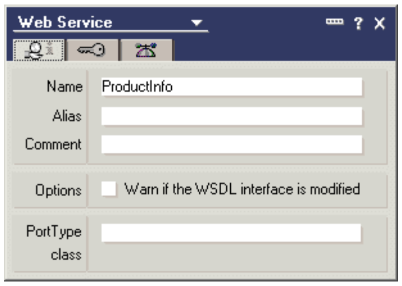
We start by excluding error and exception handling, logging, and performance and focus on the core functionality.

ITSO Electronics needs to support both Java and LotusScript.

### 3.5.1 Creating a new Web service

In this section, we demonstrate how to create a new Web service, using the ITSO Electronics application as an example:

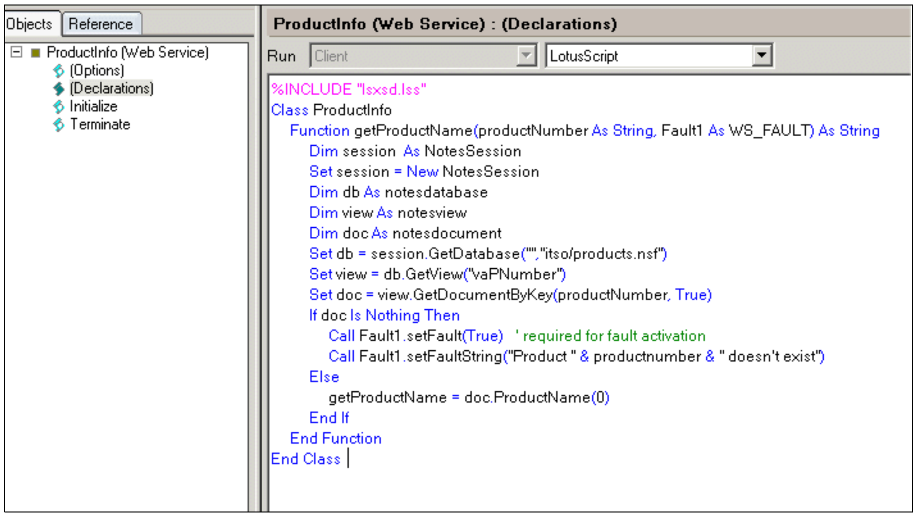
1. Open the database in Domino Designer, which in our scenario is the ITSO Web Services database. From the design list, select **Shared Code** → **Web Services**, and click **New Web Service**. In the Web Service properties window, enter `ProductInfo` as the Name of the Web Service, as shown in Figure 3-1.



| | |
|---|---|
| Name | ProductInfo |
| Alias | |
| Comment | |
| Options | ☐ Warn if the WSDL interface is modified |
| PortType class | |

*Figure 3-1 Web Service properties window*

2. To create a LotusScript Declaration, as shown in Figure 3-2, paste or type the code shown in Example 3-1 to the Declarations of the Web service.



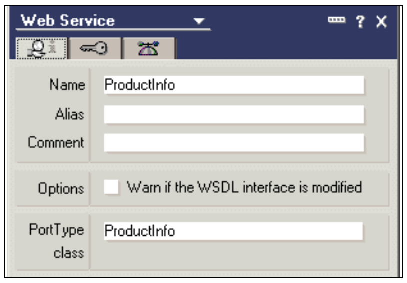
```
%INCLUDE "lsxsd.lss"
Class ProductInfo
   Function getProductName(productNumber As String, Fault1 As WS_FAULT) As String
      Dim session  As NotesSession
      Set session = New NotesSession
      Dim db As notesdatabase
      Dim view As notesview
      Dim doc As notesdocument
      Set db = session.GetDatabase("","itso/products.nsf")
      Set view = db.GetView("vaPNumber")
      Set doc = view.GetDocumentByKey(productNumber, True)
      If doc Is Nothing Then
         Call Fault1.setFault(True)   ' required for fault activation
         Call Fault1.setFaultString("Product " & productnumber & " doesn't exist")
      Else
         getProductName = doc.ProductName(0)
      End If
   End Function
End Class
```

*Figure 3-2 Source code for a simple Web service*

*Example 3-1 Code for a simple Web service*

```
%INCLUDE "lsxsd.lss"
Class ProductInfo
   Function getProductName(productNumber As String, Fault1 As WS_FAULT) As String
      Dim session  As NotesSession
      Set session = New NotesSession
      Dim db As notesdatabase
      Dim view As notesview
      Dim doc As notesdocument
      Set db = session.GetDatabase("","itso/products.nsf")
      Set view = db.GetView("vaPNumber")
      Set doc = view.GetDocumentByKey(productNumber, True)
      If doc Is Nothing Then
         Call Fault1.setFault(True)   ' required for fault activation
         Call Fault1.setFaultString("Product " & productnumber & " doesn't exist")
      Else
         getProductName = doc.ProductName(0)
      End If
   End Function
End Class
```

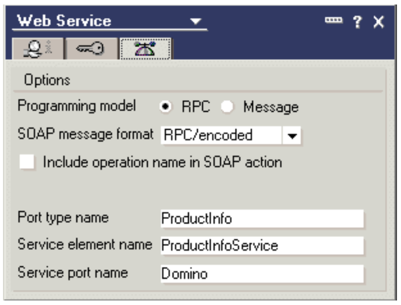
3. Re-open the Web Service properties window, and set the PortType class to `ProductInfo` before saving, as shown in Figure 3-3.



Name ProductInfo
Alias
Comment
Options ☐ Warn if the WSDL interface is modified
PortType class ProductInfo

*Figure 3-3 Web Service properties window: PortType class*

4. Go to the Advanced tab, and check that the default value for Programming model is **RPC** and for SOAP message format is **RPC/encoded**, as shown in Figure 3-4.



Programming model [✓ RPC] Message
SOAP message format RPC/encoded
Include operation name in SOAP action
Port type name ProductInfo
Service element name ProductInfoService
Service port name Domino

*Figure 3-4 Web Service properties window: Advanced tab*

5. Close the properties window, and save the Web service.

## 3.5.2 Showing the Web service in a browser

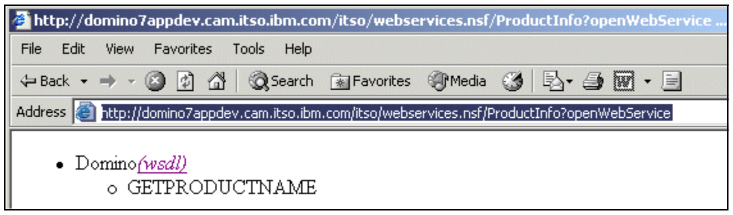
To test the newly created ProductInfo Web service, open a browser and enter the following URL:

```
http://servername/databasename.nsf/webservicename?openwebservice
```

In our ITSO Electronics application scenario, we use the following URL:

```
http://domino7appdev.cam.itso.ibm.com/itso/webservices.nsf/ProductInfo?openwebservice
```

Lotus Domino responds with the Web service's port name and the names of the operations for the port, plus a link to the WSDL document, as shown in Figure 3-5.



http://domino7appdev.cam.itso.ibm.com/itso/webservices.nsf/ProductInfo?openWebService

- Domino(wsdl)
  - GETPRODUCTNAME

*Figure 3-5 Web service opened in Web browser*

## 3.5.3 The generated WSDL document

The following Lotus Domino URL command retrieves the WSDL document for the Web service in response to an HTTP GET, using the suffix ?WSDL:

```
http://servername/databasename.nsf/webservicename?WSDL
```

For example, in our scenario, we use:

```
http://domino7appdev.cam.itso.ibm.com/itso/webservices.nsf/ProductInfo?WSDL
```

Example 3-2 on page 51 shows the resulting WSDL document.

See 3.2.1, “WSDL document” on page 45 for a detailed description of WSDL.

*Example 3-2 Code returned by the server when accessing the Web service with suffix ?WSDL*

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions xmlns="http://schemas.xmlsoap.org/wsdl/" xmlns:apachesoap="http://xml.apache.org/xml-soap"
xmlns:impl="urn:DefaultNamespace" xmlns:intf="urn:DefaultNamespace"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/" xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:wsdlsoap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="urn:DefaultNamespace">
  <wsdl:message name="GETPRODUCTNAMERequest">
    <wsdl:part name="PRODUCTNUMBER" type="xsd:string"/>
  </wsdl:message>
  <wsdl:message name="GETPRODUCTNAMEResponse">
    <wsdl:part name="GETPRODUCTNAMEReturn" type="xsd:string"/>
  </wsdl:message>
  <wsdl:portType name="ProductInfo">
    <wsdl:operation name="GETPRODUCTNAME" parameterOrder="PRODUCTNUMBER">
      <wsdl:input message="impl:GETPRODUCTNAMERequest" name="GETPRODUCTNAMERequest"/>
      <wsdl:output message="impl:GETPRODUCTNAMEResponse" name="GETPRODUCTNAMEResponse"/>
    </wsdl:operation>
  </wsdl:portType>
  <wsdl:binding name="DominoSoapBinding" type="impl:ProductInfo">
    <wsdlsoap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="GETPRODUCTNAME">
      <wsdlsoap:operation soapAction=""/>
      <wsdl:input name="GETPRODUCTNAMERequest">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" namespace="urn:DefaultNamespace"
use="encoded"/>
      </wsdl:input>
      <wsdl:output name="GETPRODUCTNAMEResponse">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" namespace="urn:DefaultNamespace"
use="encoded"/>
      </wsdl:output>
    </wsdl:operation>
  </wsdl:binding>
  <wsdl:service name="ProductInfoService">
    <wsdl:port binding="impl:DominoSoapBinding" name="Domino">
      <wsdlsoap:address
location="http://domino7appdev.cam.itso.ibm.com:80/itso/webservices.nsf/ProductInfo?OpenWebService"/>
    </wsdl:port>
  </wsdl:service>
</wsdl:definitions>
```

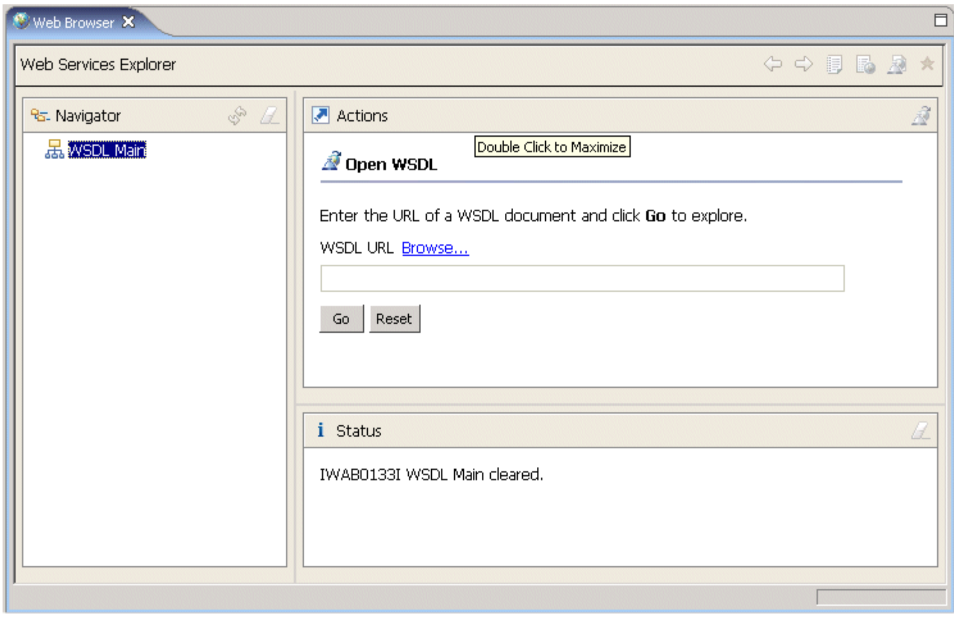
The resulting code generated by the ?WSDL suffix is complex, but it does not matter because the WSDL documents are read by machines and not read by the end user. For example, Lotus Domino Designer can import a WSDL document and create skeleton classes in LotusScript or Java. We discuss this in 3.2.3, “Imported WSDL” on page 46.

## 3.5.4 Testing the simple Web service

There are many applications that let you test and consume Web services. For the purposes of this paper, we use IBM Rational® Application Developer to test Web services. To start testing Web services using Rational Application Developer, follow these steps:

1. Open Rational Application Developer.
2. Select **Run → Launch the Web Services Explorer**.
3. Click the **WSDL page** icon (second from the right, on the upper-right side).

4. Go to the Navigator pane and select **WSDL Main**, as shown in Figure 3-6.



Enter the URL of a WSDL document and click **Go** to explore.

WSDL URL Browse...

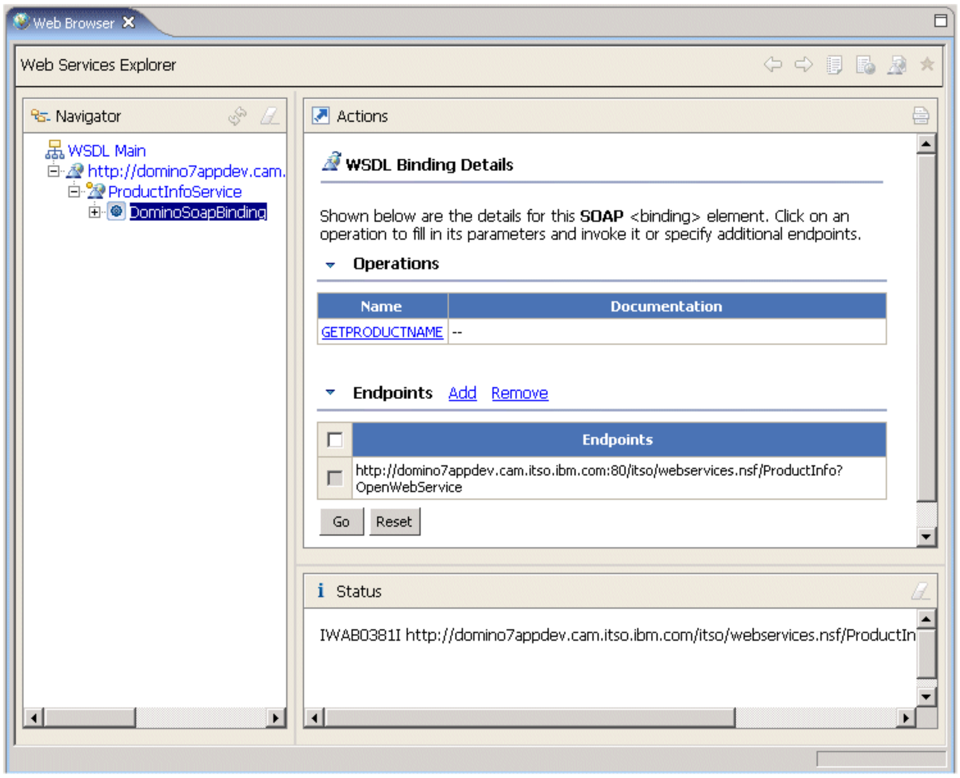
IWAB0133I WSDL Main cleared.

*Figure 3-6 Select WSDL Main from Navigator pane*

5. Go to the Open WSDL pane, and specify the WSDL URL. In our ITSO Electronics application scenario, we use the following URL:

```
http://domino7appdev.cam.itso.ibm.com/itso/webservices.nsf/ProductInfo?wsdl
```

6. Click **Go**, and the WSDL Binding Details pane opens, as shown in Figure 3-7.



Shown below are the details for this **SOAP** <binding> element. Click on an operation to fill in its parameters and invoke it or specify additional endpoints.

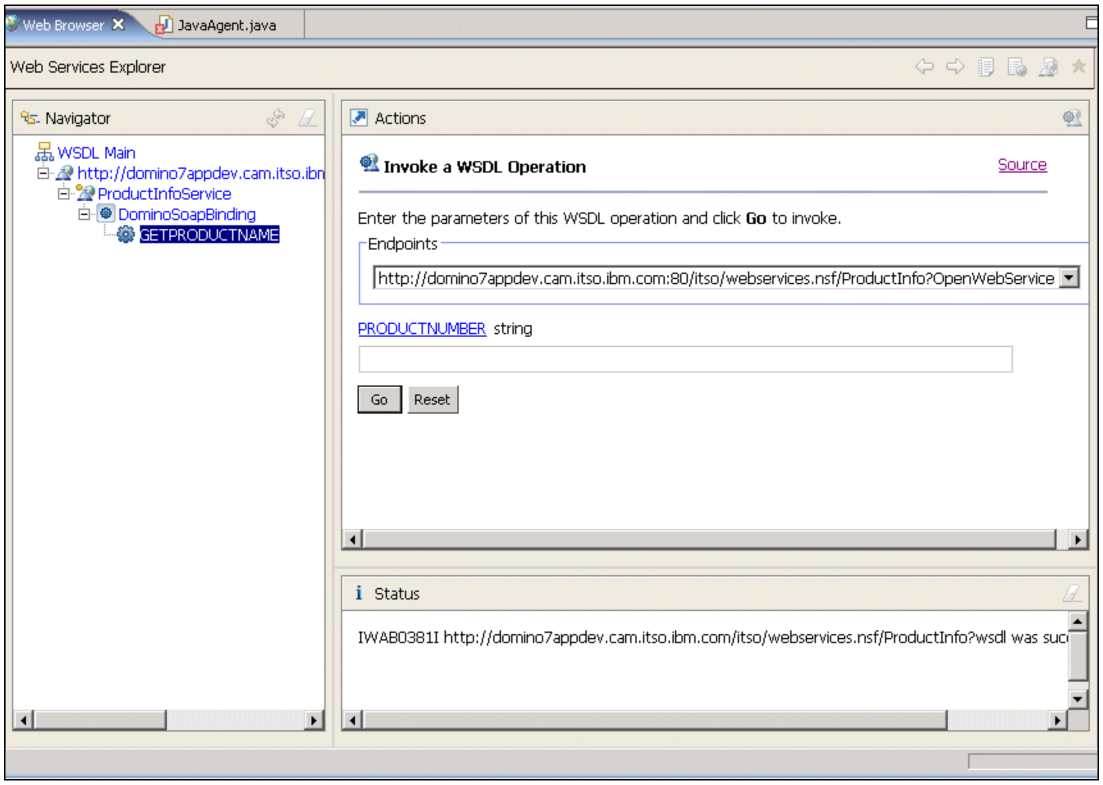
| Name | Documentation |
|---|---|
| GETPRODUCTNAME | -- |

| | Endpoints |
|---|---|
| ☐ | http://domino7appdev.cam.itso.ibm.com:80/itso/webservices.nsf/ProductInfo?OpenWebService |

IWAB0381I http://domino7appdev.cam.itso.ibm.com/itso/webservices.nsf/ProductIn

*Figure 3-7 WSDL Binding Details pane*

7. The WSDL Binding Details pane shows the operations for the SOAP element. Click an operation listed on the pane to fill in its parameters and invoke it or specify additional end points. In this scenario, select the **GETPRODUCTNAME** operation, displaying the Invoke a WSDL Operation pane, as shown in Figure 3-8.



**Invoke a WSDL Operation**

Enter the parameters of this WSDL operation and click **Go** to invoke.

Endpoints

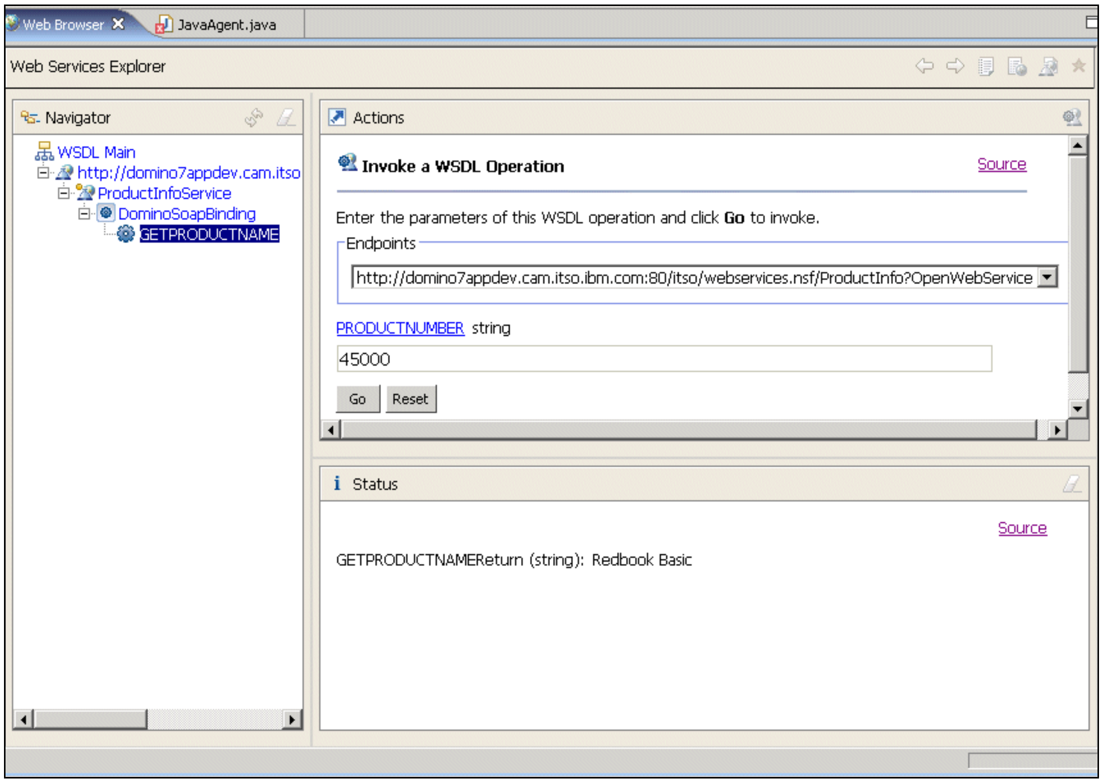
http://domino7appdev.cam.itso.ibm.com:80/itso/webservices.nsf/ProductInfo?OpenWebService

PRODUCTNUMBER string

IWAB0381I http://domino7appdev.cam.itso.ibm.com/itso/webservices.nsf/ProductInfo?wsdl was suc

*Figure 3-8 Invoke a WSDL Operation pane*

8. We enter a PRODUCTNUMBER value and click **Go**. The result is displayed on the Status pane. In our scenario, we entered 45000 as the PRODUCTNUMBER and received Redbook Basic as the result string, as shown in Figure 3-9.



Enter the parameters of this WSDL operation and click **Go** to invoke.

http://domino7appdev.cam.itso.ibm.com:80/itso/webservices.nsf/ProductInfo?OpenWebService

45000

GETPRODUCTNAMEReturn (string): Redbook Basic

*Figure 3-9 Invoke a WSDL Operation Status pane*

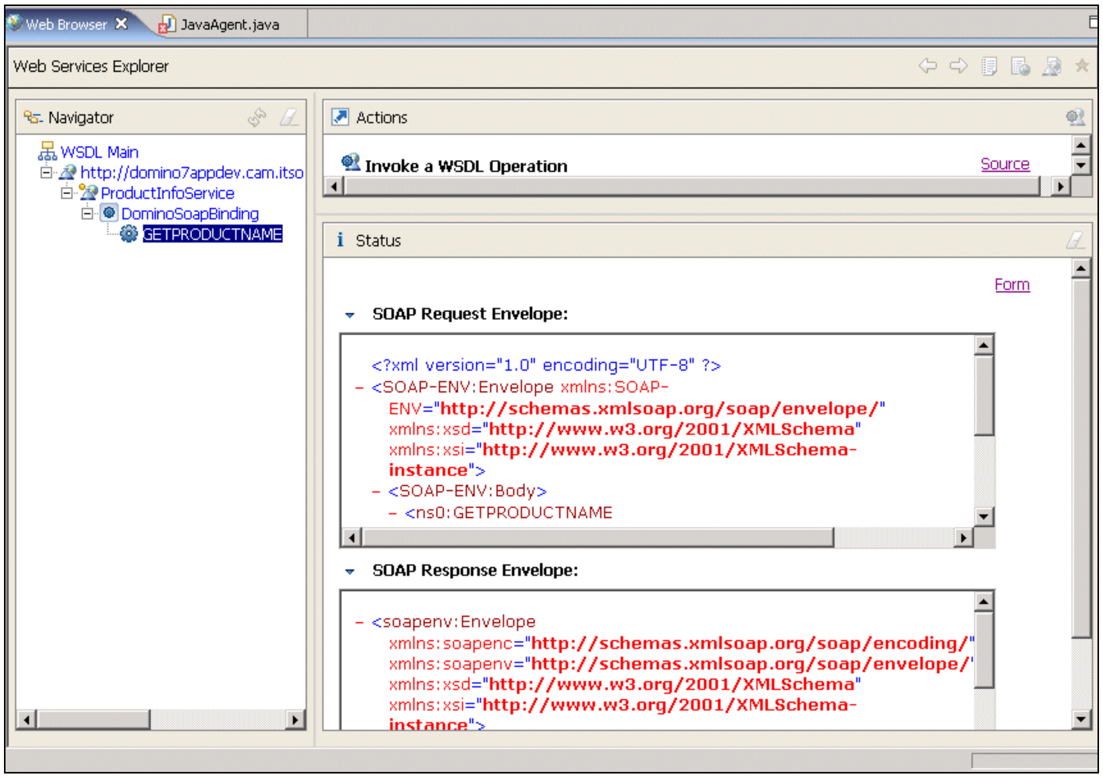
9. To see what really happened behind the scenes, click **Source** on the Status pane. Example 3-3 shows the code that was sent to the Lotus Domino server in a SOAP request envelope. The code that the Lotus Domino server responded with, by sending a SOAP response envelope, as shown in Example 3-4 on page 55, is then displayed on the Status pane, as shown in Figure 3-10 on page 55.

*Example 3-3 SOAP request envelope*

```
<?xml version="1.0" encoding="UTF-8" ?>
- <SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
- <SOAP-ENV:Body>
- <ns0:GETPRODUCTNAME xmlns:ns0="urn:DefaultNamespace"
SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <PRODUCTNUMBER xsi:type="xsd:string">45000</PRODUCTNUMBER>
  </ns0:GETPRODUCTNAME>
  </SOAP-ENV:Body>
  </SOAP-ENV:Envelope>
```

*Example 3-4   SOAP response envelope sent back from the server*

```
soapenv:Envelope xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
- <soapenv:Body>
- <ns1:GETPRODUCTNAMEResponse xmlns:ns1="urn:DefaultNamespace"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <GETPRODUCTNAMEReturn xsi:type="xsd:string">Redbook Basic</GETPRODUCTNAMEReturn>
  </ns1:GETPRODUCTNAMEResponse>
  </soapenv:Body>
  </soapenv:Envelope>
```

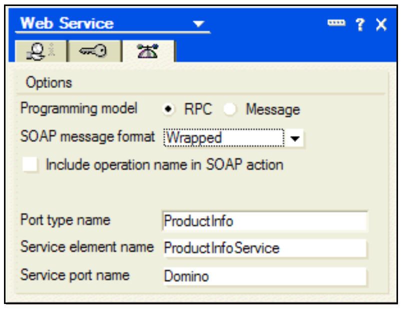


```
<?xml version="1.0" encoding="UTF-8" ?>
- <SOAP-ENV:Envelope xmlns:SOAP-
   ENV="http://schemas.xmlsoap.org/soap/envelope/"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance">
 - <SOAP-ENV:Body>
  - <ns0:GETPRODUCTNAME
```

```
- <soapenv:Envelope
   xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/'
   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance">
```

*Figure 3-10   Source code as displayed in Status pane*

10. Lastly, to see the difference in XML format, return to Lotus Domino Designer and change the SOAP message format to **Wrapped** in the advanced properties window, as shown in Figure 3-11 on page 56. Once again, invoke the Web services operation **GETPRODUCTNAME** from Rational Application Developer and the result appears, as shown in Example 3-5 on page 56 and Example 3-6 on page 56.



Programming model [✓ RPC] Message

SOAP message format: Wrapped

Include operation name in SOAP action

Port type name: ProductInfo

Service element name: ProductInfoService

Service port name: Domino

*Figure 3-11 SOAP message format Wrapped*

*Example 3-5 SOAP request envelope using SOAP message format Wrapped*

```
<?xml version="1.0" encoding="UTF-8" ?>
- <SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:q0="urn:DefaultNamespace" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
- <SOAP-ENV:Body>
- <q0:GETPRODUCTNAME>
  <PRODUCTNUMBER>45000</PRODUCTNUMBER>
  </q0:GETPRODUCTNAME>
  </SOAP-ENV:Body>
  </SOAP-ENV:Envelope>
```

*Example 3-6 SOAP response envelope when using SOAP message format Wrapped*

```
- <soapenv:Envelope xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
- <soapenv:Body>
- <ns1:GETPRODUCTNAMEResponse xmlns:ns1="urn:DefaultNamespace">
  <GETPRODUCTNAMEReturn>Redbook Basic</GETPRODUCTNAMEReturn>
  </ns1:GETPRODUCTNAMEResponse>
  </soapenv:Body>
  </soapenv:Envelope>
```

**Tip:** The Web services browser needs to get the refreshed WSDL document from the Web service. Whenever the WSDL document changes, your consumers (including the Web services browser) should be using the up-to-date version of the WSDL; otherwise, operations might not work as expected.

## 3.6 Creating a complex Web service in Lotus Domino Designer 7

The previous example demonstrated how we share primitive data types such as arrays, numbers, or in this case, strings. Next, we examine the implementation of complex data types. A business requirement of the ITSO Electronics application is to give the customers direct access to product information, and a Web service enables us to expose this data. Customers and suppliers can either look up the price for a specific product or get a collection of all available products.

In order to grant customers access to product information, we need to share product information such as:

- Product name
- Price
- Number
- Description

Example 3-7 shows the data representation of a Product object. This object will be used as the complex data type in the Web service that we are about to create in the remainder of this section. This example employs the Java programming language.

*Example 3-7 Product.java code*

```
public class Product
{
   private String name;
   private int number;
   private String description;
   private double price;
   public String getName(){ return name; }
   public int getNumber(){ return number; }
   public double getPrice(){ return price; }
   public String getDescription() { return description; }

   public void setName(String newName)
   {
         name = newName;
   }
   public void setNumber(int newNumber)
   {
      number = newNumber;
   }
   public void setDescription(String newDescription)
   {
      description = newDescription;
   }
   public void setPrice(double newPrice)
   {
      price = newPrice;
   }
}
```

In 3.5.1, “Creating a new Web service” on page 48, we create a Web service using a primitive data type. In this example, we use the same steps to create a Web service for our Product object, which will provide our partners with detailed product information. The following steps summarize how to create the Web service:

1. Open the ITSO Web Services database in Domino Designer, and select **Shared Code → Web Services**. Click **New Web Service**.
2. Use `ProductService` as the Name of the Web service, and specify `ProductService` as the PortType class.
3. In the (Options) pane, switch to **Java**.
4. Click **New Class**, and replace the default code with the code supplied in Example 3-8 on page 58, which is the ProductService.java code.

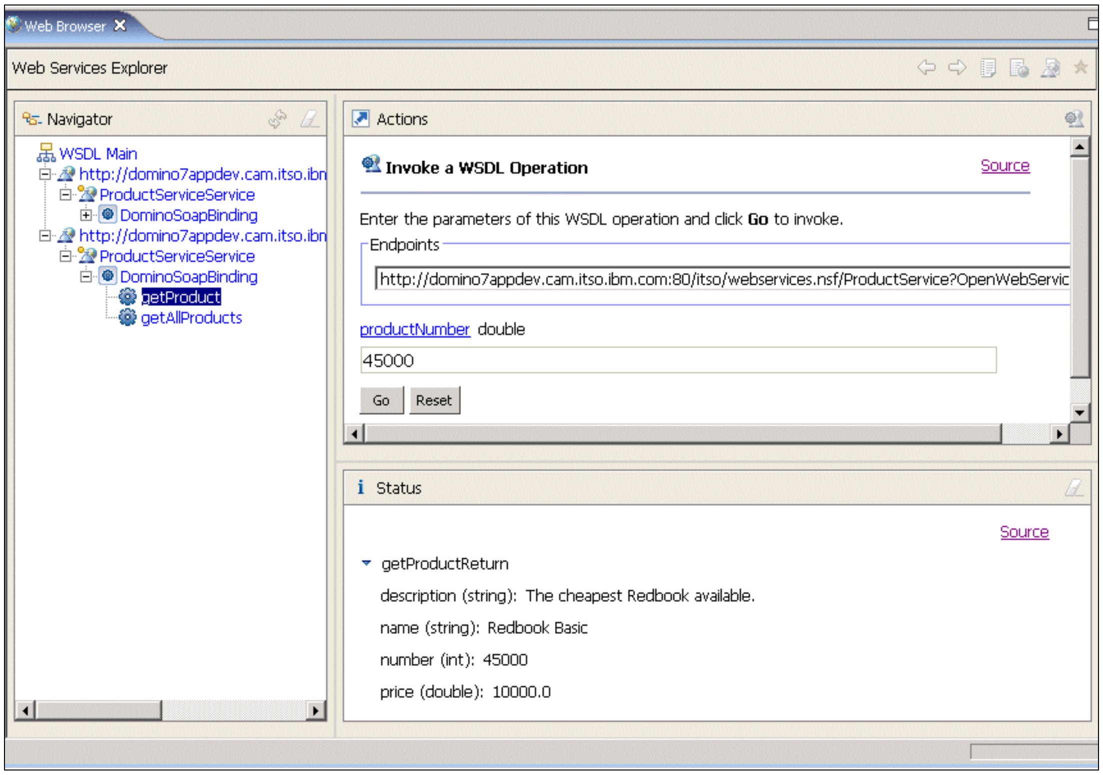
5. Click **New Class** again, and paste in the code supplied in Example 3-7 on page 57, which is the Product.java code.
6. Save the Web service.

*Example 3-8 ProductService.java*

```
import lotus.domino.*;
import lotus.domino.types.*;
import java.util.*;

public class ProductService
{
   private Session session = WebServiceBase.getCurrentSession();
   public Object[] getAllProducts()
   {
      Product[] result = null;
      Database db = null;
      View view = null;
      Product product = null;
      ArrayList tempList = new ArrayList();
      try
      {
         db = getProductDb();
         view = db.getView("vaPNumber");
          Document doc2, doc = view.getFirstDocument();
         while(doc != null)
         {
            product = getProductFromDocument(doc);
            tempList.add(product);
            doc2 = doc;
            doc = view.getNextDocument(doc);
            doc2.recycle();
         }
         view.recycle();
         db.recycle();
      }catch(NotesException e){e.printStackTrace();}
      return tempList.toArray();
   }
   private Product getProductFromDocument(Document doc)
   {
      Product product = null;
      try
      {
         String name = doc.getItemValueString("ProductName");
         String description = doc.getItemValueString("ProductDescription");
         double price = doc.getItemValueDouble("ProductPrice");
         int productNumber = doc.getItemValueInteger("productNumber");
         product = new Product();
         product.setNumber(productNumber);
         product.setName(name);
         product.setPrice(price);
         product.setDescription(description);
      }
      catch(Exception e)
      {}
      return product;
   }
   private Database getProductDb()
   {
      Database db = null;
```

```
        try{
            db = session.getDatabase("","itso\\products.nsf");
        }catch(NotesException e){}
        return db;
    }
    public Product getProduct(double productNumber)
    {
        Product product = null;
        try
        {
            Database db = getProductDb();
            View view = db.getView("vaPNumber");
             Document doc = view.getDocumentByKey(new Double(productNumber), true);
            if(doc != null)
            {
                product = getProductFromDocument(doc);
                doc.recycle();
                doc = null;
            }
            view.recycle();
            db.recycle();
        }catch(NotesException e){}
        return product;
    }

}
```

We can consume this new Web service called ProductService from Rational Application Developer, as shown in Figure 3-12.



Invoke a WSDL Operation

Enter the parameters of this WSDL operation and click **Go** to invoke.

Endpoints

http://domino7appdev.cam.itso.ibm.com:80/itso/webservices.nsf/ProductService?OpenWebServic

productNumber double

45000

getProductReturn

description (string): The cheapest Redbook available.

name (string): Redbook Basic

number (int): 45000

price (double): 10000.0

*Figure 3-12 Invoking the WSDL operation getProduct*

Example 3-9 shows the SOAP response envelope that we get back from the Lotus Domino server for a specific product number that we use to invoke the WSDL operation getProduct.

*Example 3-9 SOAP response envelope from the server*

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
- <soapenv:Body>
- <ns1:getProductResponse xmlns:ns1="urn:DefaultNamespace"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <getProductReturn href="#id0" />
  </ns1:getProductResponse>
- <multiRef xmlns:ns2="urn:DefaultNamespace"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/" id="id0" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" xsi:type="ns2:Product">
  <description xsi:type="xsd:string">This is the basic Redbook offering.</description>
  <name xsi:type="xsd:string">Basic Redbook</name>
  <number xsi:type="xsd:int">20501</number>
  <price xsi:type="xsd:double">102.0</price>
  </multiRef>
  </soapenv:Body>
  </soapenv:Envelope>
```

The ProductService that we created has another operation called getAllProducts, which sends a collection of product objects as an array. Example 3-10 illustrates this, which has a SOAP message format of RPC/encoded.

*Example 3-10 SOAP response envelope*

```
- <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
- <soapenv:Body>
- <ns1:getAllProductsResponse xmlns:ns1="urn:DefaultNamespace"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
- <getAllProductsReturn xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
soapenc:arrayType="xsd:anyType[9]" xsi:type="soapenc:Array">
  <item href="#id0" />
  <item href="#id1" />
  <item href="#id2" />
  <item href="#id3" />
  <item href="#id4" />
  <item href="#id5" />
  <item href="#id6" />
  <item href="#id7" />
  <item href="#id8" />
  </getAllProductsReturn>
  </ns1:getAllProductsResponse>
- <multiRef xmlns:ns2="urn:DefaultNamespace"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/" id="id1" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" xsi:type="ns2:Product">
  <description xsi:type="xsd:string">Redpaper mid-tier line available at newsagents and
department stores.</description>
  <name xsi:type="xsd:string">Redpaper Deluxe</name>
  <number xsi:type="xsd:int">10002</number>
  <price xsi:type="xsd:double">2299.0</price>
  </multiRef>
```

```
- <multiRef xmlns:ns3="urn:DefaultNamespace"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/" id="id8" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" xsi:type="ns3:Product">
  <description xsi:type="xsd:string">The best Redbook money can buy.</description>
  <name xsi:type="xsd:string">Redbook Plus</name>
  <number xsi:type="xsd:int">45500</number>
  <price xsi:type="xsd:double">50000.0</price>
  </multiRef>

and so on.......

- <multiRef xmlns:ns10="urn:DefaultNamespace"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/" id="id6" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" xsi:type="ns10:Product">
  <description xsi:type="xsd:string">The ultimate Redbook experience.</description>
  <name xsi:type="xsd:string">Redbook Supreme</name>
  <number xsi:type="xsd:int">20564</number>
  <price xsi:type="xsd:double">1000.0</price>
  </multiRef>
  </soapenv:Body>
  </soapenv:Envelope>
```

## 3.7 Exporting a WSDL document

WSDL documents are exported to provide the customer with the information about how to invoke our Web services. In many cases, the tools for creating Web services clients can auto-generate code from a WSDL document.

To export the WSDL document, perform the following steps:

1. Open the ITSO Web Services database in Domino Designer, and select **Shared Code** → **Web Services**.
2. Select **ProductService** and click **Export WSDL**.
3. A WSDL file, which is an XML file, is generated. In our scenario, we use the .wsdl extension to identify our exported WSDL file, as shown in Figure 3-13 on page 62.

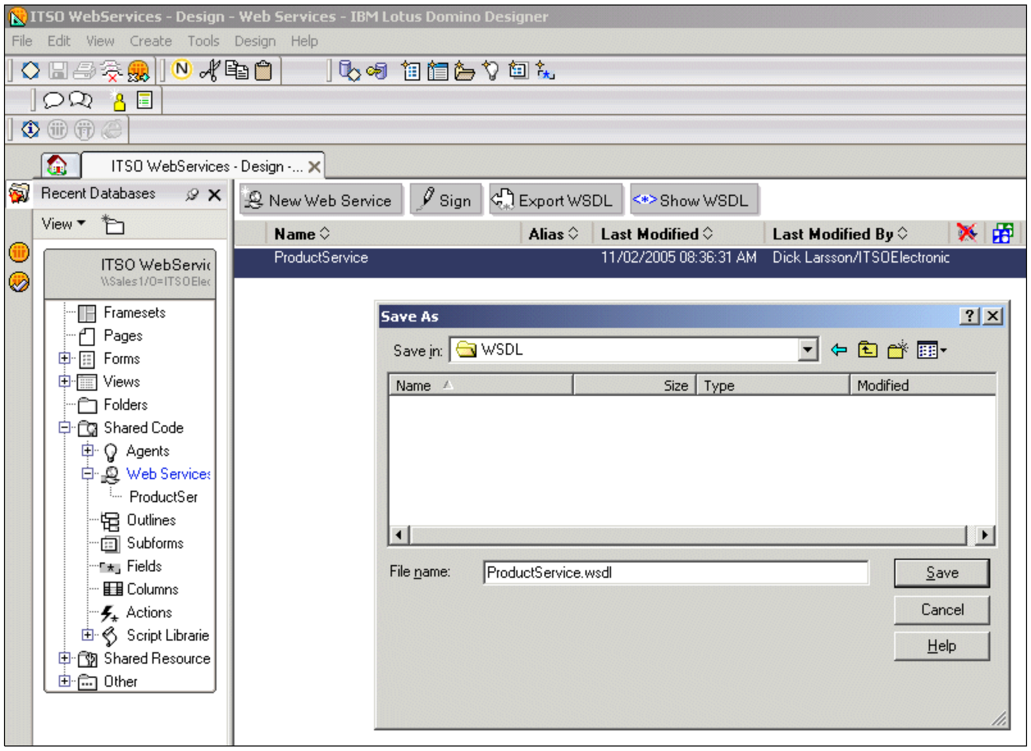


*Figure 3-13 Export WSDL document*

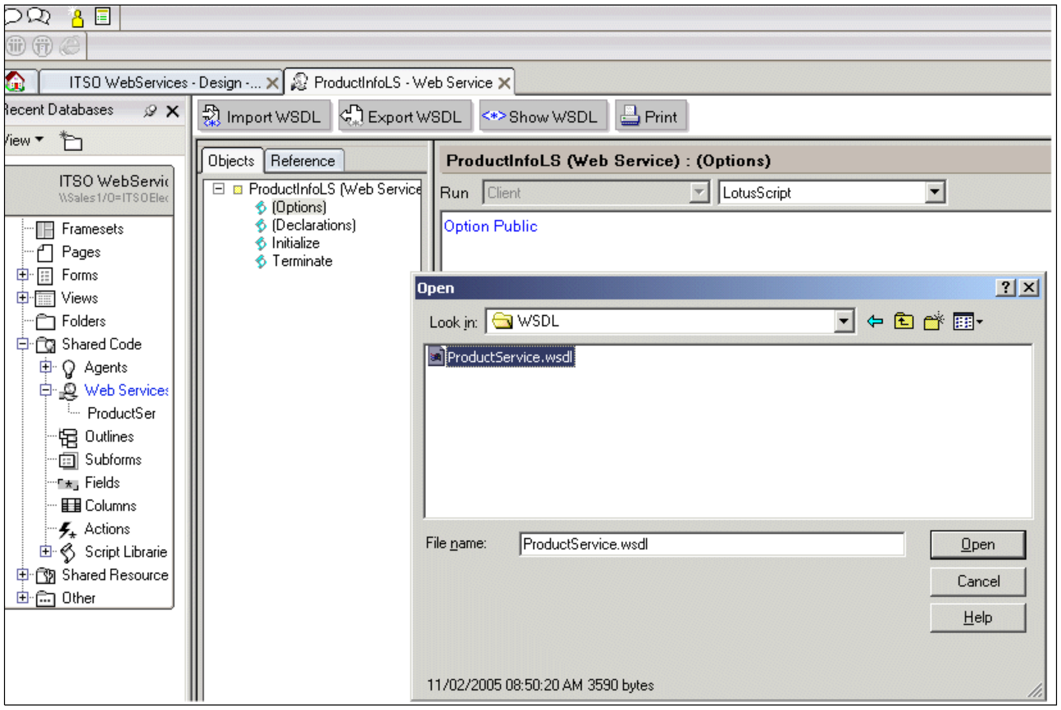
## 3.8 Importing a WSDL document

If you import a WSDL document into a Web service, an implementation class is generated based on the WSDL content. The public prototype methods, functions, and subs in the class correspond to the Web service operations defined for the first <port> element of the first <service> element in the WSDL document. Other classes or types referenced in the prototype methods might also be generated from the WSDL document, along with their public data members. If you change a prototype interface, the corresponding WSDL document changes when the Web service is saved. You can monitor and prevent such changes with the “Warn if the WSDL interface is modified” property.

ITSO Electronics business partners, and possibly customers, in return, can supply a WSDL document that the company can use to create Web services. You can import an existing WSDL document to generate a skeleton Web service. The skeleton code corresponds to the Web service definition. You then add implementation code.

To import a WSDL document, perform the following steps:

1. Open the ITSO Web Services database in Domino Designer, and select **Shared Code** → **Web Services**.
2. Click **Import WSDL**, and select the WSDL file to be imported. In our scenario, we import the file that we exported in the previous section, as shown in Figure 3-14 on page 63.



Run: Client — LotusScript

```
Option Public
```

Look in: WSDL

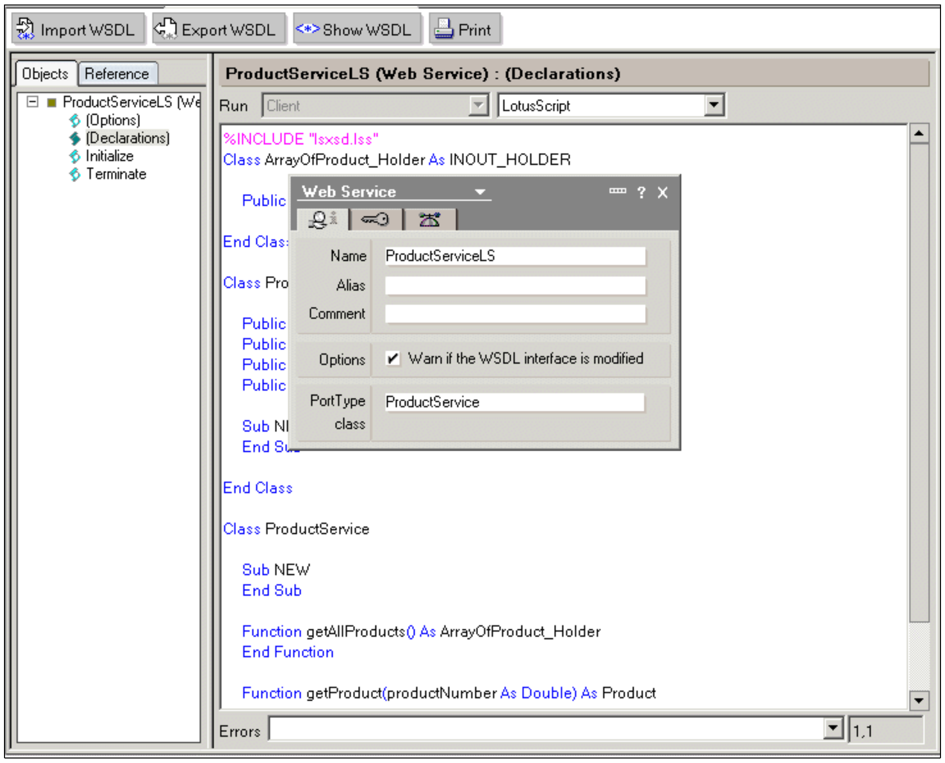
ProductService.wsdl

File name: ProductService.wsdl

11/02/2005 08:50:20 AM 3590 bytes

Figure 3-14   Import WSDL document

3. Importing the WSDL document generates skeleton code, which in our scenario is generated as LotusScript, as shown in Example 3-11 on page 64. We then open the Web Service properties window, and select the **Warn if the WSDL interface is modified** option, as shown Figure 3-15.



Run: Client — LotusScript

```
%INCLUDE "lsxsd.lss"
Class ArrayOfProduct_Holder As INOUT_HOLDER

  Public

End Clas

Class Pro

  Public
  Public
  Public
  Public

  Sub NI
  End Su

End Class

Class ProductService

  Sub NEW
  End Sub

  Function getAllProducts() As ArrayOfProduct_Holder
  End Function

  Function getProduct(productNumber As Double) As Product
```

| | |
|---|---|
| Name | ProductServiceLS |
| Alias | |
| Comment | |
| Options | ✓ Warn if the WSDL interface is modified |
| PortType class | ProductService |

Figure 3-15   Import WSDL document: Web Services properties window

*Example 3-11 Skeleton code generated after importing the WSDL document*

```
%INCLUDE "lsxsd.lss"
Class ArrayOfProduct_Holder As INOUT_HOLDER

Public Value() As Product

End Class

Class Product

Public description As XSD_STRING
Public name As XSD_STRING
Public number As Long
Public price As Double

Sub NEW
End Sub

End Class

Class ProductService

Sub NEW
End Sub

Function getAllProducts() As ArrayOfProduct_Holder
End Function

Function getProduct(productNumber As Double) As Product
End Function

End Class
```

## 3.9 Exception and error handling in Web services

Operations can also return errors to the caller, either through a fault subclass that corresponds to some <wsdl:fault> message specified for the operation in the WSDL document, or through direct use of the WS_FAULT base class (LotusScript), or the lotus.domino.types.Fault or java.lang.Exception classes (Java).

**Note:** Any faults for a LotusScript service implementation method (that is, any WS_FAULT base class or its subclasses) must appear at the end of the method argument list.

Faults for a Java service implementation method appear in the conventional Java throws clause for the method.

See 3.12, "Consuming Web services using LS2J" on page 71 for information about how to handle faults using LotusScript.

## 3.10 Web services security

Threats to Web services as it is implemented in the ITSO Electronics application involve threats to the host system, the application, and the entire network infrastructure. To secure Web services, a range of XML-based security mechanisms are needed to solve problems related to authentication, role-based access control, distributed security policy enforcement, and message layer security that accommodate the presence of intermediaries.

At this time, there are no broadly-adopted specifications for Web services security. As a result, developers can either build up services that do not use these capabilities or can develop ad-hoc solutions that might lead to interoperability problems.

Web services implementations might require point-to-point or end-to-end security mechanisms, or both, depending upon the degree of threat or risk. Traditional, connection-oriented, point-to-point security mechanisms might not meet the end-to-end security requirements of Web services. However, security is a balance of assessed risk and cost of countermeasures. Depending on the implementer's risk tolerance, point-to-point, transport-level security can provide enough security countermeasures.

From the perspective of the ITSO Electronics application architecture, there are three fundamental concepts related to security: the resources that must be secured; the mechanisms by which these resources are secured (that is, policy guards); and policies, which are machine-processable documents describing constraints on these resources. For a thorough discussion of the security considerations and implications of implementing Web services, review the *Web Service Architecture* guide, available at:

http://www.w3.org/TR/2004/NOTE-ws-arch-20040211

A Lotus Domino 7 Web service has the same security capabilities as an agent. The security of a Web service is set in the Security tab of the Web service properties window. The following list describes the options available in the Security tab:

- Run as web user: The Java or LotusScript code runs with the effective user name of the user invoking the Web service.
- Run on behalf of: The Java or LotusScript code runs on the authority of a specified user. However, the signer of the Web service must have unrestricted rights on the server or have the right to run on behalf of anyone.
- Set runtime security level (one of the following):
  - Do not allow restricted operations.
  - Allow restricted operations.
  - Allow restricted operations with full administration rights.
- Default access for this web service:
  - All readers and above. You can select or clear this option.
  - Enumerated. If the All readers and above is cleared, you can select who you want to have default access.
- Allow Public Access users to use this web service. Allows default access to those who have public access to documents in the database.

## 3.11 Consuming Web services using Java

At the time of this writing, Lotus Domino 7.0 provides, but does not natively consume, Web services. However, a secondary business requirement for ITSO Electronics is to gain access to pricing data beyond the constraints of Lotus Domino, which dictates that the application consume Web services. In our scenario, ITSO Electronics has chosen to use the open source product Apache SOAP to consume Web services using Java. Apache SOAP (Simple Object Access Protocol) is an implementation of the SOAP submission to W3C. It is based on, and supersedes, the IBM SOAP4J implementation. The follow-on project is Apache Axis, available at:

http://ws.apache.org/axis/

To illustrate consuming Web services in our scenario, we create a new ITSO Electronics application database called WS Consumer. This database will contain an agent that will consume the complex Web service that we created earlier.

**Note:** Web services can be added to any Lotus Notes and Domino databases. However, for the purposes of this paper, all Web services are added to separate databases in the ITSO Electronics application called webservices.nsf and wsconsumer.nsf.

The following steps show how we create an agent that consumes the Web service:

1. Download the latest version of Apache SOAP from:

   http://ws.apache.org/soap/

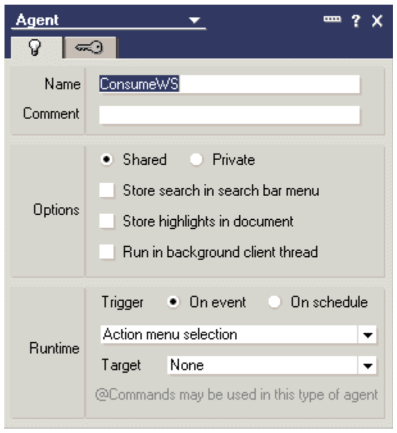
2. Extract the downloaded file into a temp directory, which in our scenario is c:\soap.
3. Export the complex Web service called ProductService (in the Web Services database) to a WSDL document. The exported WSDL document contains all the information needed to consume the Web service.
4. Open the WSDL document in a text editor. The WSDL document contains information describing the complex data type returned by the service, as shown in Example 3-12. In this case, the complex data type is called Product. The complexType element shows the properties for the Product object, which are description, name, number, and price.

*Example 3-12 WSDL type for Product*

```
<wsdl:types>
  <schema targetNamespace="urn:DefaultNamespace" xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
   <import namespace="http://schemas.xmlsoap.org/soap/encoding/"/>
   <complexType name="Product">
    <sequence>
     <element name="description" nillable="true" type="xsd:string"/>
     <element name="name" nillable="true" type="xsd:string"/>
     <element name="number" type="xsd:int"/>
     <element name="price" type="xsd:double"/>
    </sequence>
   </complexType>
   <complexType name="ArrayOfProduct">
    <complexContent>
     <restriction base="soapenc:Array">
      <attribute ref="soapenc:arrayType" wsdl:arrayType="impl:Product[]"/>
     </restriction>
    </complexContent>
   </complexType>
  </schema>
 </wsdl:types>
```

5. Open the new WS Consumer database in Domino Designer, select **Shared Code** → **Agents**, and click **New agent**.

6. Set the Name of the agent as `ConsumeWS`, and select **None** for the Runtime Target, as shown in Figure 3-16.

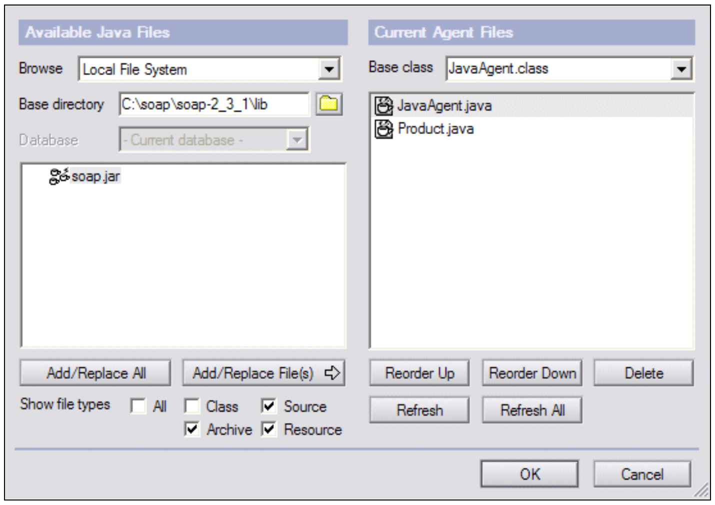

Agent

Name: ConsumeWS
Comment:
Options: [✓ Shared] Private
Store search in search bar menu
Store highlights in document
Run in background client thread
Runtime: Trigger [✓ On event] On schedule
Action menu selection
Target: None
@Commands may be used in this type of agent

*Figure 3-16 ConsumerWS Agent properties window*

7. Close the Agent properties window, and select **Java** for Action.
8. We need to create the JavaBean class representing the Product complex data type as specified in the WSDL document. To make it easier, we take advantage of the import WSDL functionality to will automatically generate the JavaBean class. We now go back to the design list, and select **Shared Code** → **Web Services** and click **New Web Service**.
9. Close the Web Service properties window, and ensure that (Options) is set to Java. Click **Import WSDL**, and select the exported WSDL document that was created earlier. JavaBeans™ classes are automatically created with the import.
10. Select and copy the Java code for Product.java.
11. Go back to the agent and click **New Class**. Then, replace the Untitled.java class with the copied Product.java code, as shown in Example 3-13 on page 68.
12. Save the ConsumeWS agent and ensure that the code compiles successfully.

    Click **Edit Project**, and the Organize Java Agent Files opens, as shown in Figure 3-17. Go to the correct SOAP LIB library directory, which in our case is c:\soap\soap-2_3_1\llib. Select the **Archive** file type, click **Add/Replace File(s)**, and click **OK**.


Available Java Files

Browse: Local File System
Base directory: C:\soap\soap-2_3_1\lib
Database: - Current database -
soap.jar

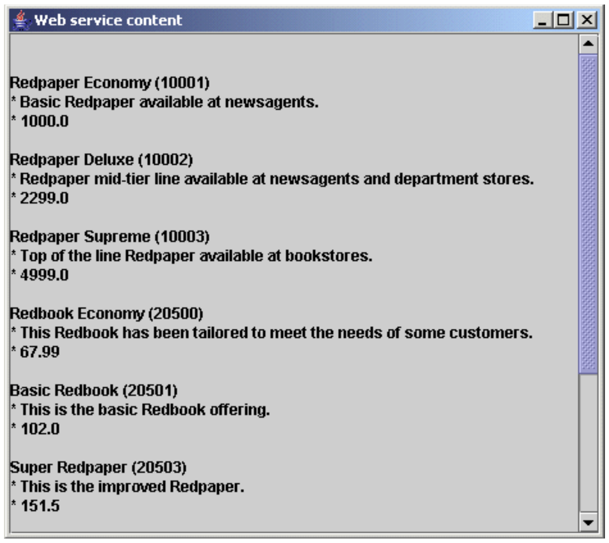
Show file types: All; Class; [✓ Source]; [✓ Archive]; [✓ Resource]

Current Agent Files

Base class: JavaAgent.class
JavaAgent.java
Product.java

*Figure 3-17 Organize Java Agent Files window*

13. You need to replace the default JavaAgent.java code with your own code to consume Web services, which in our scenario is in Example 3-14 on page 70. Ensure that the JavaAgent.java class has a URL and soapAction variables. These variables must point to the location of your application.
14. Save the ConsumeWS agent.
15. Go back to the new Web Service and close it without saving.
16. Open the Lotus Notes client, and select the WS Consumer database. Go to **Actions** → **ConsumeWS** to trigger the newly created agent. The agent displays a collection of Product objects consumed by the Web service in a Java window, as shown in Figure 3-18.



```
Redpaper Economy (10001)
* Basic Redpaper available at newsagents.
* 1000.0

Redpaper Deluxe (10002)
* Redpaper mid-tier line available at newsagents and department stores.
* 2299.0

Redpaper Supreme (10003)
* Top of the line Redpaper available at bookstores.
* 4999.0

Redbook Economy (20500)
* This Redbook has been tailored to meet the needs of some customers.
* 67.99

Basic Redbook (20501)
* This is the basic Redbook offering.
* 102.0

Super Redpaper (20503)
* This is the improved Redpaper.
* 151.5
```

*Figure 3-18 Java window displaying the Web service result*

If the window does not open, select **File** → **Tools** → **Show Java Debug Console** for more information.

*Example 3-13 Code for class Product*

```
public class Product  {
   private java.lang.String description;
   private java.lang.String name;
   private int number;
   private double price;

   public Product() {
   }

   public java.lang.String getDescription() {
      return description;
   }

   public void setDescription(java.lang.String description) {
      this.description = description;
   }

   public java.lang.String getName() {
      return name;
   }

   public void setName(java.lang.String name) {
```

```
   this.name = name;
}

public int getNumber() {
   return number;
}

public void setNumber(int number) {
   this.number = number;
}

public double getPrice() {
   return price;
}

public void setPrice(double price) {
   this.price = price;
}

private java.lang.Object __equalsCalc = null;
public synchronized boolean equals(java.lang.Object obj) {
   if (!(obj instanceof Product)) return false;
   Product other = (Product) obj;
   if (obj == null) return false;
   if (this == obj) return true;
   if (__equalsCalc != null) {
      return (__equalsCalc == obj);
   }
   __equalsCalc = obj;
   boolean _equals;
   _equals = true &&
      ((this.description==null && other.getDescription()==null) ||
       (this.description!=null &&
        this.description.equals(other.getDescription()))) &&
      ((this.name==null && other.getName()==null) ||
       (this.name!=null &&
        this.name.equals(other.getName()))) &&
      this.number == other.getNumber() &&
      this.price == other.getPrice();
   __equalsCalc = null;
   return _equals;
}

private boolean __hashCodeCalc = false;
public synchronized int hashCode() {
   if (__hashCodeCalc) {
      return 0;
   }
   __hashCodeCalc = true;
   int _hashCode = 1;
   if (getDescription() != null) {
      _hashCode += getDescription().hashCode();
   }
   if (getName() != null) {
      _hashCode += getName().hashCode();
   }
   _hashCode += getNumber();
   _hashCode += new Double(getPrice()).hashCode();
   __hashCodeCalc = false;
   return _hashCode;
```

```
    }

}
```

*Example 3-14   Code for consuming a Web Service using Java*

```
import lotus.domino.*;
import java.io.*;
import java.util.*;
import java.net.*;
import org.w3c.dom.*;
import org.apache.soap.util.xml.*;
import org.apache.soap.*;
import org.apache.soap.encoding.*;
import org.apache.soap.encoding.soapenc.*;
import org.apache.soap.rpc.*;
import javax.swing.*;

public class JavaAgent extends AgentBase
{
    public void NotesMain()
    {
        try
        {
            Session session = getSession();
            AgentContext agentContext = session.getAgentContext();

            URL url = new URL("http://domino7appdev.cam.itso.ibm.com/itso/webservices.nsf/ProductService?WSDL");
            String soapAction ="http://domino7appdev.cam.itso.ibm.com/itso/webservices.nsf/ProductService?WSDL";

            Call call = new Call();
            BeanSerializer beanSerializer = new BeanSerializer();
            SOAPMappingRegistry smRegistry = new SOAPMappingRegistry();

            smRegistry.mapTypes(Constants.NS_URI_SOAP_ENC, new QName(
                    "urn:DefaultNamespace", "Product"), Product.class, beanSerializer,
                    beanSerializer);

            String targetNamespace = "urn:DefaultNamespace";
            call.setSOAPMappingRegistry(smRegistry);
            call.setTargetObjectURI(targetNamespace);
            call.setMethodName("getAllProducts");
            call.setEncodingStyleURI(Constants.NS_URI_SOAP_ENC);

            Response resp = null;
            try
            {
                resp = call.invoke(url, soapAction);
            } catch (SOAPException e)
            {
                e.printStackTrace();
                System.err.println("SOAP Exception code: " + e.getFaultCode() + " message: " + e.getMessage());
                return;
            }

            if (!resp.generatedFault())
            {
                Parameter ret = resp.getReturnValue();
                Product[] products = (Product[]) ret.getValue();
                buildGUI(products);
            }
           else
            {
                Fault fault = resp.getFault();
                System.err.println("SOAP Fault,  " + fault.getFaultCode() + "   " + fault.getFaultString());
            }

        } catch (Exception e)
        {
            e.printStackTrace();
        }
    }
    private void buildGUI(Product[] products)
    {
           StringBuffer sb = new StringBuffer();
           String newLine = "<br>";
           for(int i = 0; i  < products.length; i++)
           {
           sb.append(newLine + newLine);
           sb.append(products[i].getName() + " (" + products[i].getNumber() + ")" + newLine);
           sb.append( " * " + products[i].getDescription());
```

```
            sb.append(newLine + " * " + products[i].getPrice());
         }
            JFrame frame = new JFrame("Web service content");
            frame.getContentPane().add(new JScrollPane(new JLabel("<html>" + sb.toString() + "</html>")));
            frame.pack();
            frame.setVisible(true);
      }
}
```

## 3.12 Consuming Web services using LS2J

In this section, we show you:

- How to consume a Web service using LotusScript
- How to use LS2J to call Java code from a LotusScript agent
- How to work with exceptions or errors sent back from a Web service
- Authentication and Web services

### 3.12.1 About LS2J

LS2J is the interface that enables data to transfer from the Java data type to the LotusScript data type and enables LotusScript to execute Java object methods. With LS2J, LotusScript can create Java objects as though they are native to the LotusScript environment.

In this scenario, we add a LotusScript agent to the WS Consumer database. The agent uses Java code from a script library to consume a Web service.

The Web service we consume can do one of two things:

- Return the name of the current user
- Return a WS_FAULT object to illustrate how Web services can send error codes back to the client

### 3.12.2 Creating the Web service

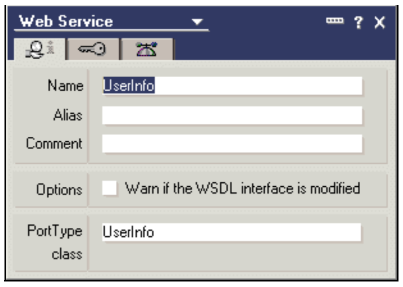
The following steps illustrate how we create the Web service using LS2J. This section describes the creation of the Web service, and the following sections describe how the Java script library is created, and how that script library is used in our LotusScript agent through LS2J. Perform the following steps:

1. Open the Web Services database in Domino Designer.
2. Click the **New Web Service** button.
3. Verify that the language is set to **LotusScript**.
4. Go to (Declarations) and type in the code shown in Example 3-15.

*Example 3-15 LS2J Web Service code*

```
Dim session As NotesSession
Class UserInfo
        Function getUserName ( Fault1 As WS_FAULT)
                Set session = New NotesSession
                Dim doc As NotesDocument
                Set doc = session.DocumentContext

                If  Cint(Rnd() * 10) > 5 Then
                        getUserName =  doc.remote_user(0)
                Else
                        Call Fault1.setFault(True)   ' required for fault activation
             Call Fault1.setFaultString("SUCCESS, We send a FAULT on purpose")
                End If

        End Function
End Class
```

5. Open the properties window for the Web service and set the Name and PortType class to `UserInfo`, as shown Example 3-19.



Name: UserInfo
Alias:
Comment:
Options: Warn if the WSDL interface is modified
PortType class: UserInfo

*Figure 3-19 LS2J Web Service UserInfo properties window*

6. Save and close the Web service.

## 3.12.3 Creating the script library for consuming the Web service using Java

We now need to create a script library containing the Java class that will consume the Web service and make it available for our LotusScript agent through LS2J. To set up the Java class:

1. Open the WS Consumer database in Domino Designer.
2. Click **Shared Code** in the navigation pane, and click **New Java Library**.
3. In the newly opened window, click **Edit Project** and select the Apache **soap.jar** file, as shown in Figure 3-20 on page 73. If you do not have the Apache soap.jar file, refer to 3.11, “Consuming Web services using Java” on page 65 for more information about downloading Apache SOAP.

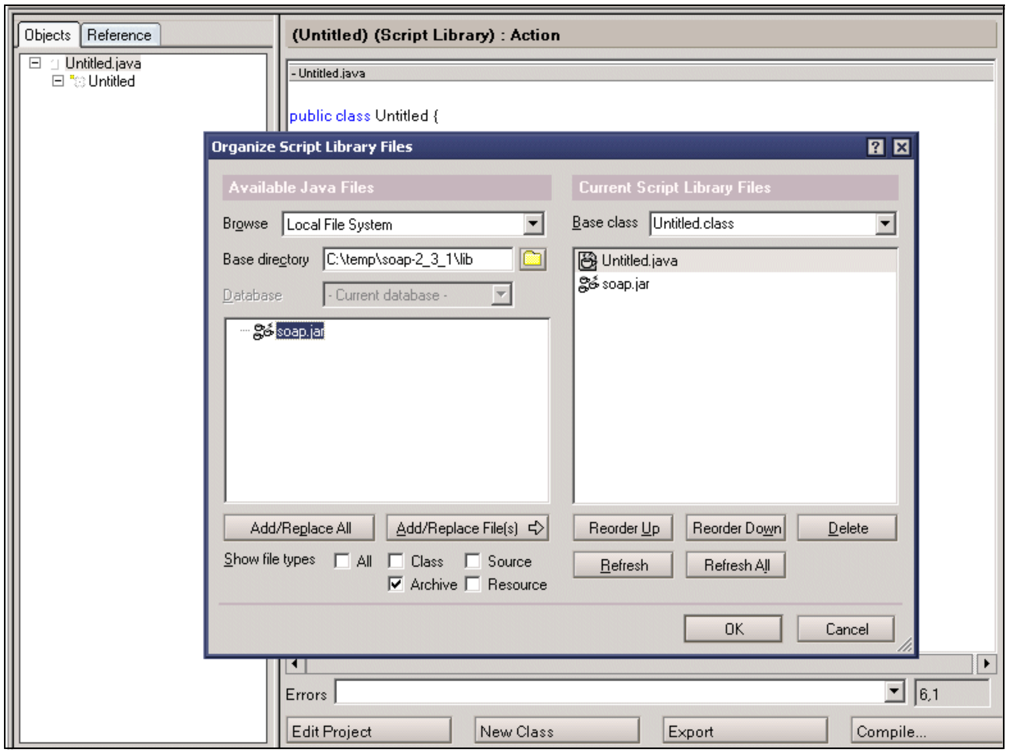


*Figure 3-20 Import Java archive to be used with LS2J*

4. Save the script library as ConsumeWSLibrary.
5. Replace the default code for the script library with the code shown in Example 3-16.

**Important:** The following lines from Example 3-16 must be updated with the URL to your database:

```
URL url = new URL("http://domino7appdev.cam.itso.ibm.com/itso/webservices.nsf/ProductService?WSDL");
String soapAction ="http://domino7appdev.cam.itso.ibm.com/itso/webservices.nsf/ProductService?WSDL";
```

*Example 3-16 Java code for consuming Web service to be used by LS2J*

```
import java.io.*;
import java.util.*;
import java.net.*;
import org.w3c.dom.*;
import org.apache.soap.util.xml.*;
import org.apache.soap.*;
import org.apache.soap.encoding.*;
import org.apache.soap.encoding.soapenc.*;
import org.apache.soap.rpc.*;
import org.apache.soap.transport.http.SOAPHTTPConnection;


public class WSConsumer
{
    public String getValue(String userName, String password)
    {
        try
        {

            URL url = new URL("http://domino7appdev.cam.itso.ibm.com/itso/webservices.nsf/UserInfo?WSDL");
            String soapAction ="http://domino7appdev.cam.itso.ibm.com/itso/webservices.nsf/UserInfo?WSDL";

            Call call = new Call();
            SOAPHTTPConnection hc = new SOAPHTTPConnection();
          hc.setUserName(userName);
          hc.setPassword(password);
            call.setSOAPTransport(hc);
            SOAPMappingRegistry smRegistry = new SOAPMappingRegistry();
            String targetNamespace = "urn:DefaultNamespace";
            call.setSOAPMappingRegistry(smRegistry);
            call.setTargetObjectURI(targetNamespace);
            call.setMethodName("getUserName");
            call.setEncodingStyleURI(Constants.NS_URI_SOAP_ENC);

            Response resp = null;
            try
            {
                resp = call.invoke(url, soapAction);
            } catch (SOAPException e)
            {
                e.printStackTrace();
                return ("SOAP Exception code: " + e.getFaultCode() + " message: " + e.getMessage());
            }

            if (!resp.generatedFault())
            {
                Parameter ret = resp.getReturnValue();
             return ret.getValue() + "";
            }
           else
            {
                Fault fault = resp.getFault();
                return ("SOAP Fault,  " + fault.getFaultCode() + "   " + fault.getFaultString());
            }
        } catch (Exception e)
        {
            e.printStackTrace();
        }
        return null;
    }

}
```

## 3.12.4 Creating the LotusScript agent

Finally, we create the LotusScript agent and use the newly created Java script library through LS2J:

1. In the Navigation pane, select **Shared Code** → **Agents**.
2. Click **New Agent**. In the properties window for the new agent, set the Name to `LotusScriptConsumeWS` and set Runtime Target to **None**. Close the properties window.
3. Change the language to LotusScript.
4. In Declarations, replace the code with the code from Example 3-17.
5. In Initialize, replace the code from Example 3-18.

**Note:** Update the line value `= myObject.getValue("The UserName", "The Password")` with a valid user name and password.

*Example 3-17 Declaration code for the LotusScript LS2J agent*

```
Option Public
Use "ConsumeWSLibrary"
Uselsx "*javacon"
```

*Example 3-18 Initialize code for the LotusScript LS2J agent*

```
Sub Initialize
   Dim mySession As JavaSession
   Dim myClass As JavaClass
   Dim myObject As JavaObject

   Set mySession = New JavaSession ()
   Set myClass = mySession.GetClass("WSConsumer")
```

```
    Set myObject = myClass.CreateObject
    Dim value As String
    value = myObject.getValue("The UserName", "The Password")
    Msgbox value
End Sub
```

6. Save and close the agent.
7. In the Lotus Notes client, run the newly created agent.
   If you get a prompt message, as shown in Figure 3-21, this might be because you have not supplied correct user name and password.

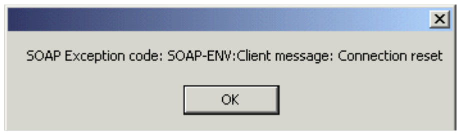

SOAP Exception code: SOAP-ENV:Client message: Connection reset

*Figure 3-21 Error message while consuming Web service using LS2J*

If the agent was able to consume your Web service, you will get a prompt message, as shown in Figure 3-22.

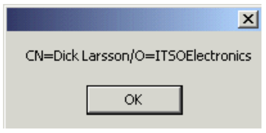

CN=Dick Larsson/O=ITSOElectronics

*Figure 3-22 Web service successfully consumed using LS2J*

### 3.12.5 Authentication and Web services

In Lotus Notes and Domino 7, a Web service has the same security capabilities as an agent. In Example 3-19, we use HTTP basic authentication to secure our Web service. Apache SOAP lets us send the user name and password with the SOAP request by adding an instance of the class SOAPHTTPConnection to the Call object.

*Example 3-19 Basic authentication with Apache SOAP*

```
Call call = new Call();
SOAPHTTPConnection hc = new SOAPHTTPConnection();
hc.setUserName(userName);
hc.setPassword(password);
call.setSOAPTransport(hc);
```

## 3.13 Which style of WSDL should I use?

A WSDL document contains the XML code that describes a Web service. A WSDL binding describes how the service is bound to a messaging protocol, particularly the SOAP messaging protocol. A WSDL SOAP binding can be either a remote procedure call (RPC) style binding or a document style binding. A SOAP binding can also have an encoded use or a literal use. This offers the Lotus Domino application developer four style/use models from which to choose:

- RPC/encoded
- RPC/literal
- Document/literal

- Wrapped

The terminology “RPC versus document” might be confusing, because these terms imply that the RPC style should be used for RPC programming models and that the document style should be used for document or messaging programming models. However, this is not the case, because the style has nothing to do with a programming model. It merely dictates how to translate a WSDL binding to a SOAP message. Likewise, the terms encoded and literal are only meaningful for the WSDL-to-SOAP mapping.

4

# Implementing new design elements in the ITSO Electronics application

The new features in Lotus Domino Designer 7 build on the themes of tighter integration with evolving Web standards, increased interoperability with other IBM Software standards, and improved ease of use for applications developers and end users.

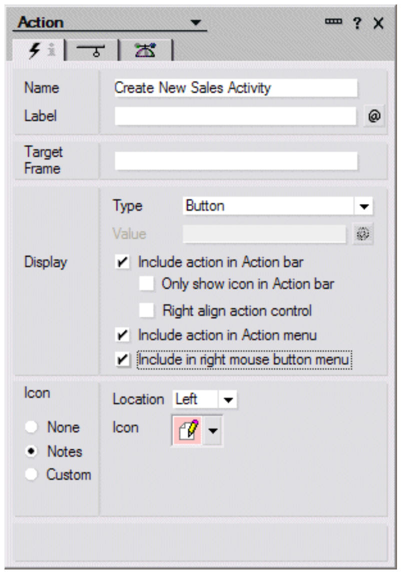
In this chapter, we illustrate how the new features of Lotus Domino Designer 7 are used to enhance the ITSO Electronics application. We focus on the following new features:

- View actions in right-click menus
- AutoSave form property
- Shared columns
- Enhanced runtime customizations for views and folders
- Additions to formula language
- Additions to LotusScript, COM, and OLE classes
- Additions to Java and CORBA classes

## 4.1 View actions in right-click menus

Application developers can now add actions to right-click menus in views and folders. Right-clicking is intended as a shortcut, a way to get to functions that users use routinely. Right-clicking is technically no different from clicking a button on the Action bar; it operates on the selected documents on the view. From a user interface perspective, its advantage is that you can select a document and carry out an action on it with a small mouse movement. It is useful for cases where the user is likely to want to act on a single document based on information they can see in a view, for example, reply or forward, file to a frequently used folder, delete, or highlight for later attention. Adding an action to the right-click menu can be used as an alternative to, or in conjunction with, including the action in the Action bar and Action menu. Both new and existing actions can be added to the right-click menu.

In the ITSO Electronics application, the “Sales Activity By Customer” view has a Create New Sales Activity action button. To enhance the application and increase the productivity of our users, we update the Display option of the action and select **Include in right mouse button menu**, as shown in Figure 4-1.



| | |
|---|---|
| Name | Create New Sales Activity |
| Label | |
| Target Frame | |
| Display | Type: Button |
| | Value |
| | ✓ Include action in Action bar |
| | ☐ Only show icon in Action bar |
| | ☐ Right align action control |
| | ✓ Include action in Action menu |
| | ✓ Include in right mouse button menu |
| Icon | None [✓ Notes] Custom |
| | Location: Left |
| | Icon |

*Figure 4-1 Include in right mouse button menu option*

Right-clicking the view displays the right-click menu and will now include the Create New Sales Activity action, as shown in Figure 4-2 on page 79. We also added the other view actions to the right-click menu.

**Note:** The right-click menu is not accessible for users with disabilities. All actions made available in the right-click menu should also be available from the Actions menu.

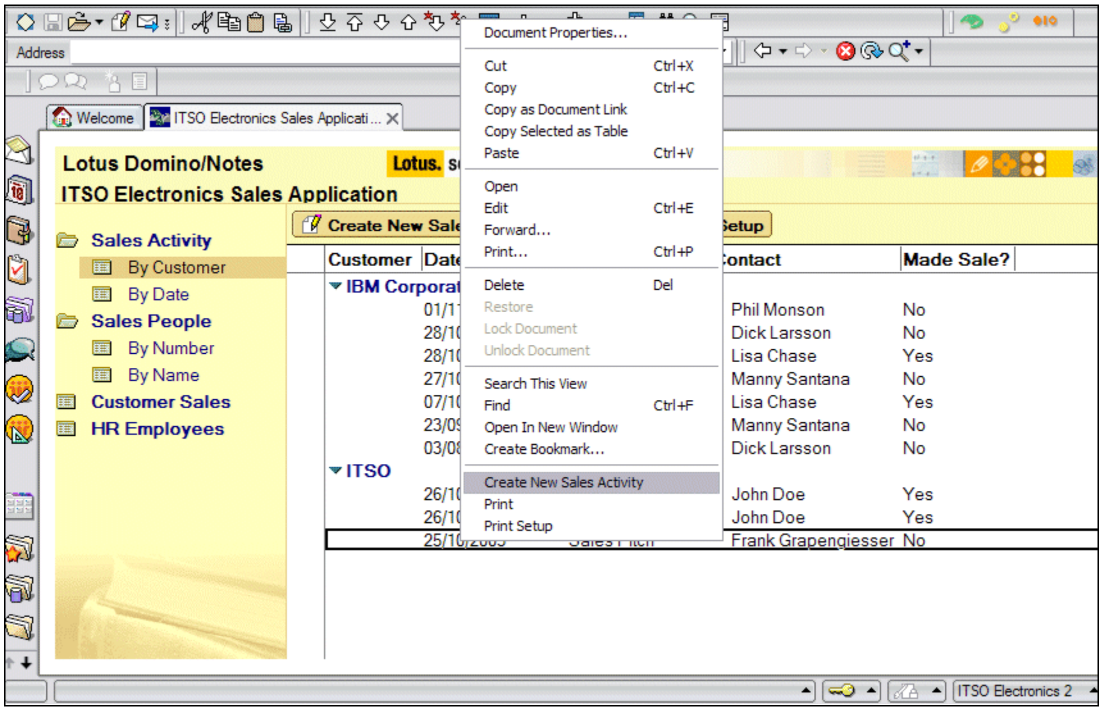


*Figure 4-2 Action included in right-click menu*

## 4.2 AutoSave form property

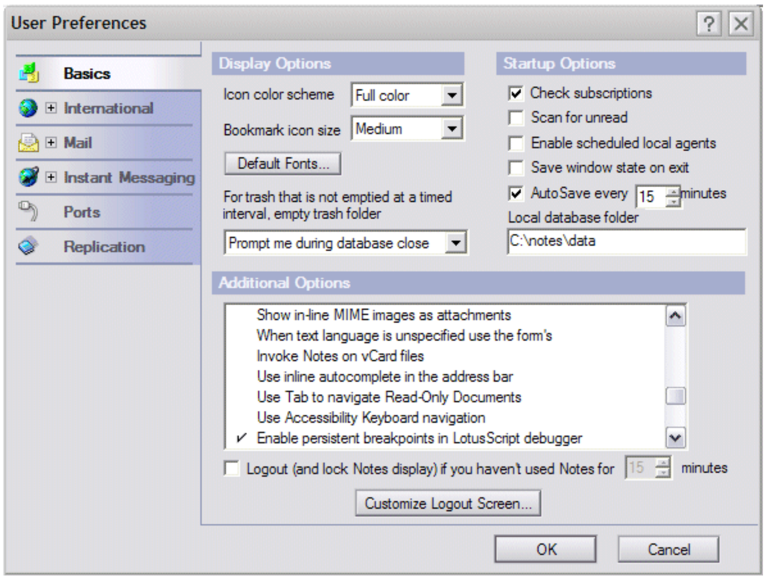
AutoSave is a Lotus Notes client feature by which Lotus Notes documents are automatically saved to a local database at regular intervals as determined by the user. In the event of a software failure, workstation failure, or power outage, the user can recover the work that was done prior to the crash or power loss. As diligent consultants, we recommended that the ITSO Electronics company follow the instructions in the subsequent sections and consider protecting their Lotus Domino applications using the new AutoSave feature.

### 4.2.1 Enabling AutoSave on the Lotus Notes client

AutoSave is, by default, not enabled when Lotus Notes 7 is installed. To enable AutoSave, the option must be selected in the Lotus Notes Client User Preferences:

1. Select **File** → **Preference** → **User Preferences**.
2. On the Basics pane, under Startup Options, enable **AutoSave every 15 minutes**, as shown in Figure 4-3 on page 80.
3. (Optional) Increase or decrease the AutoSave interval.
4. Click **OK**.

**Note:** The default AutoSave time is 15 minutes. The time interval has a range of 1 to 999 minutes.



Icon color scheme: Full color
Bookmark icon size: Medium
For trash that is not emptied at a timed interval, empty trash folder: Prompt me during database close

☑ Check subscriptions
☐ Scan for unread
☐ Enable scheduled local agents
☐ Save window state on exit
☑ AutoSave every 15 minutes
Local database folder: C:\notes\data

Show in-line MIME images as attachments
When text language is unspecified use the form's
Invoke Notes on vCard files
Use inline autocomplete in the address bar
Use Tab to navigate Read-Only Documents
Use Accessibility Keyboard navigation
✓ Enable persistent breakpoints in LotusScript debugger

☐ Logout (and lock Notes display) if you haven't used Notes for 15 minutes

*Figure 4-3 Enabling AutoSave in User Preferences*

The AutoSave options in the User Preferences window correspond to three variables in the Lotus Notes client's NOTES.INI file, as shown in Table 4-1.

*Table 4-1 Variables for NOTES.INI*

| NOTES.INI variable | User Preferences setting |
|---|---|
| `AUTO_SAVE_ENABLE=n` | AutoSave is disabled when `n=0`. |
| `AUTO_SAVE_INTERVAL=<mins>` | `<mins>` sets the time interval between AutoSaves. |
| `AUTO_SAVE_USER,<Abbreviated User Name>=<Database name relative to data directory>` | Resolves naming conflicts with AutoSave databases. |

## 4.2.2 Enabling AutoSave on a form

Lotus Domino Designer 7 takes advantage of the new AutoSave feature by adding the ability to enable a form in a Lotus Notes application to support the AutoSave feature. If users activate AutoSave in their Lotus Notes preferences, as shown in the previous section, documents that users create or edit using an AutoSave-enabled form will be saved to a special local encrypted database. To activate AutoSave for a form, open the form in Domino Designer and enable the **Allow Autosave** option in the Form properties window, as shown in Figure 4-4 on page 81. When AutoSave is enabled in the Lotus Notes client, the status bar will indicate when a document is being autosaved.

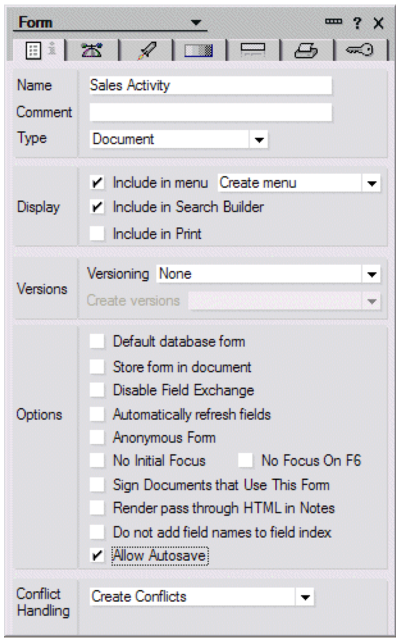


*Figure 4-4 Allow Autosave option in Form properties window*

Application developers should make sure that AutoSave works properly with their application when they enable AutoSave on the application forms. Many Lotus Notes and Domino applications have code that create items in their Save or Postsave events, or perform validations that are checked at load time. Because validations and form event code do not execute while autosaving a document, the autosaved document might fail to load after recovery. It might then be necessary to add code in the Queryopen event to put default values for those particular items to ensure that the documents will load properly after being recovered.

The $AutosaveRecovered item can be used to handle documents that are recovered. $AutoSaveRecovered is available for recovered documents up until the PostOpen event and is deleted after the event. For example, a form might contain Queryopen code to prevent a document from being opened in edit mode if the “Status” field doesn't equal “Draft”. If the user has changed the status to “Publish”, AutoSave takes place, and then the user’s computer crashes, the user still needs to be able to open that document in edit mode even though the status seems to have the wrong value. Note the following code:

```
Sub Queryopen(Source As Notesuidocument, Mode As Integer, Isnewdoc As Variant, Continue As 
Variant)
   Dim doc As NotesDocument
   If Mode And Not Isnewdoc Then
      Set doc = Source.Document
      If doc.Status(0) <> "Draft" Then
         Continue = False
         Messagebox "Only Draft documents may be edited.", _
MB_ICONSTOP, "Edit Error"
      End If
   End If
End Sub
```

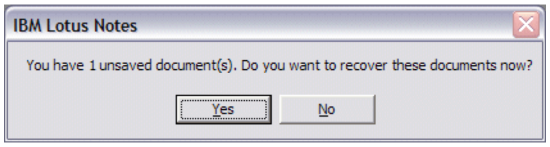
You might instead write:

```
If doc.Status(0) <> "Draft" And Not doc.HasItem("$AutosaveRecovered ") Then
```

The $DontAutosave item can be used to disable AutoSave on individual documents, preventing AutoSave on that particular document. The document can be AutoSaved again by removing $DontAutosave.

## 4.2.3 Recovering documents saved with the AutoSave feature

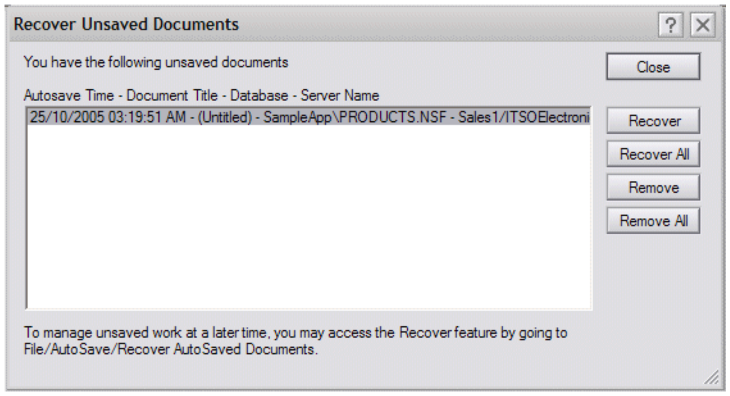
AutoSaved documents can be recovered either immediately after startup, or at any other time after startup. At startup after a failure or power outage, and after the user authenticates, the user is prompted to recover unsaved work, as shown in Figure 4-5.


IBM Lotus Notes

You have 1 unsaved document(s). Do you want to recover these documents now?

*Figure 4-5 Unsaved work prompt*

If the user clicks **Yes**, the Recover Unsaved Documents window opens, listing all the documents that can be recovered, as shown in Figure 4-6.


Recover Unsaved Documents

You have the following unsaved documents

Autosave Time - Document Title - Database - Server Name

25/10/2005 03:19:51 AM - (Untitled) - SampleApp\PRODUCTS.NSF - Sales1/ITSOElectroni

To manage unsaved work at a later time, you may access the Recover feature by going to File/AutoSave/Recover AutoSaved Documents.

*Figure 4-6 Recover Unsaved Documents window*

Table 4-2 describes the function of each of the four buttons in the window shown in Figure 4-6.

*Table 4-2 Recover Unsaved Documents options*

| Option | Description |
|---|---|
| Recover | Recovers the selected document and removes it from the list. |
| Recover All | Recovers all documents without prompting for each one. |
| Remove | Removes the selected document from the AutoSave database. |
| Remove All | Removes all documents from the AutoSave database. |

If the user clicks **No**, the AutoSaved documents can be recovered later by selecting **File** → **AutoSave** → **Recover AutoSaved Documents**. The Recover Unsaved Documents window opens, and users can recover or delete documents as described earlier.

The document cannot be recovered if a replica of the document's database cannot be found. The document is left as-is in the AutoSave database, and recovery can be tried again at a later time. If the document has been modified since the last AutoSave, the user is prompted and given the option to overwrite using the AutoSaved document or keep the modified document. If the user chooses to recover the AutoSaved document, it is opened in Lotus Notes in edit mode. AutoSave will not automatically save the document on the user's behalf after recovery. If a document is recovered from the AutoSave database, it is also deleted from the database, which assists in keeping the size of the AutoSave database manageable.

### 4.2.4 Using AutoSave manually

**Note:** This option only works if the form has been enabled for AutoSave.

Occasionally, users might be in the middle of editing a document and need to save the document immediately. They have the option of saving the current document to the AutoSave database by selecting **File** → **AutoSave** → **Autosave Now**.

### 4.2.5 Accessing the AutoSave database

The AutoSave database is located in the user's Data directory. When the AutoSave database is created, it uses a similar naming convention as mail files. The database name is prepended with as_. In our scenario application, the user Manny Santana will have an AutoSave database name of as_msantana.nsf. If a database with that name already exists, 1 is appended to the name, and so on, until a unique name is created for the AutoSave database.

With the AutoSave database naming convention, conflicts will occur for user names such as John Doe, Jane Doe, James Doe, and so forth. To ensure that the right AutoSave database is used for the current user, after the database is created, an entry is appended to the user's NOTES.INI file:

```
AUTO_SAVE_USER,<Abbreviated User Name>=<Database name relative to data directory>
```

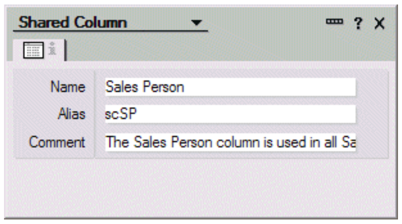
In our sample application, this entry will be added to Manny Santana's NOTES.INI file:

```
AUTO_SAVE_USER,Manny Santana/ITOSElectronics=as_msantana.nsf
```

When user starts Lotus Notes, the NOTES.INI file is checked to see whether an entry corresponding to the AutoSave database exists. If an entry exists, the database name specified in the entry is used as the AutoSave database. If the user does not have access to the specified AutoSave database, the user receives an error and AutoSave is disabled. If the database specified does not exist, a new database is created using the name specified in the entry. If there is no entry in the NOTES.INI file with the user's name, an entry is created in the NOTES.INI file, and a database is created using the naming convention previously mentioned.

## 4.3 Shared columns

Lotus Domino Designer 7 software provides the ability to create shared columns for use in views and folders. A shared column has its own design element but, once inserted into a view, is part of the view. The shared column design element has the same column properties and appearance as an unshared column in a view or folder. Furthermore, the shared column design element has properties specific to shared columns. The Shared Column properties window, as shown in Figure 4-7 on page 84, specifies the name of the shared column (required), an alias (optional), and a comment (optional).



Name: Sales Person
Alias: scSP
Comment: The Sales Person column is used in all Sa

Figure 4-7 Shared Column properties window

A view or folder containing a shared column is backward compatible for access from the Web and Lotus Notes. Shared columns can be used in DB2 Query Views, as discussed in 2.3, “Query Views” on page 18. Shared columns, like shared fields, are not multilingual elements.

Using shared columns helps increase developer productivity by allowing the creation of a common column for insertion in multiple views and folders in the same database, thereby eliminating the need to create the same column multiple times. This new feature creates a single point for propagating changes to all views using the same column.

## 4.3.1 Creating a shared column

To create a new shared column, select **Columns** under Shared Code. Click **New Shared Column**, or select **Create** → **Design** → **Shared Column**. You can also copy and paste an existing shared column design element.

To create a new shared column based on the design of an existing column in a view, open the view, select the column, and select **Create** → **Copy As Shared Column**. Alternatively, select the column, right-click to access the menu and select **Copy As Shared Column**, as shown in Figure 4-8.

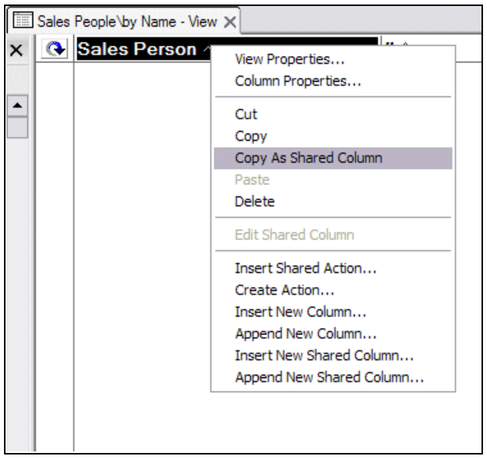


Figure 4-8 Using the right-click menu to create a new shared column

A window opens, as shown in Figure 4-9 on page 85, to name the new shared column. This action creates a new shared column and leaves the existing view column as is. The existing column is totally independent of the new shared column, so if the user wants to inherit design changes from the new shared column, the user must delete the existing view column and insert the new shared column into the view to replace that unshared column.

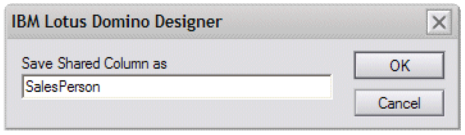
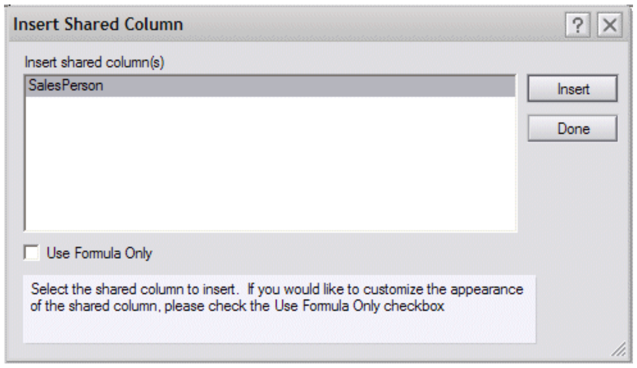


*Figure 4-9 Naming a new Shared Column*

In the ITSO Electronics application, the “Sales People By Name” view has a Sales Person column. We need to display the same information in the “Sales People By Number” view. Therefore, the Sales Person column is used to create a new shared column called Sales Person as shown earlier. Views affected by shared columns are accessible by prior releases of the Lotus Notes client.

## 4.3.2 Using a shared column

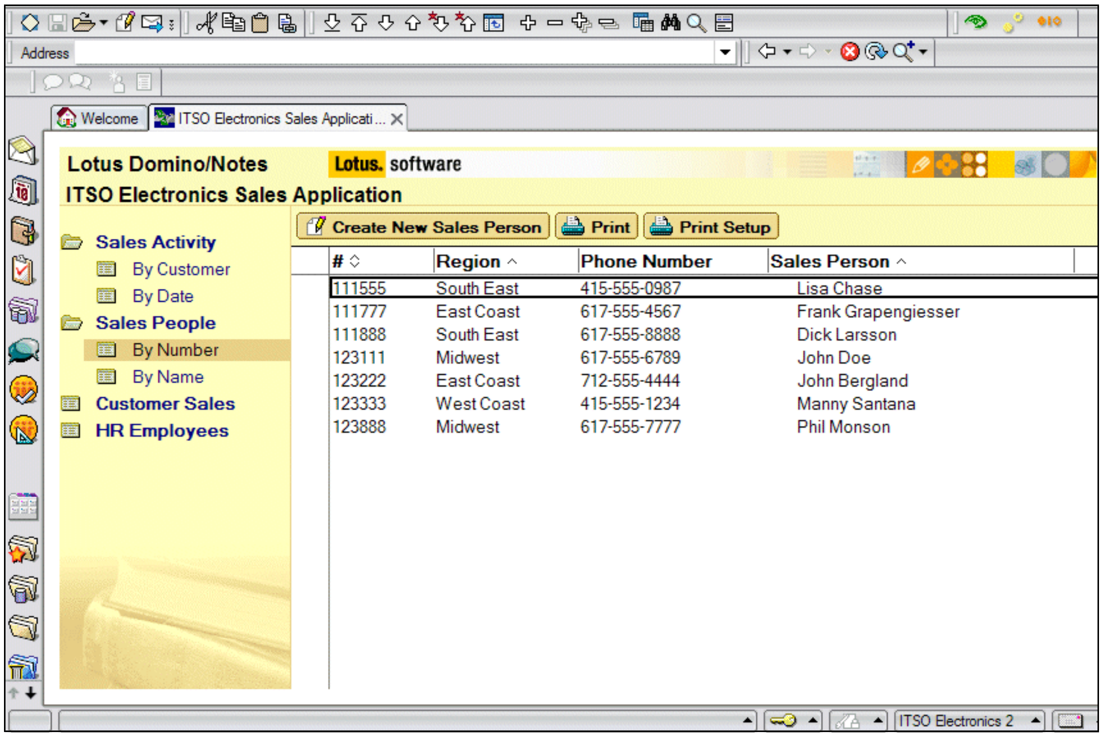
To use a shared column in a view, open the view and select **Create** → **Insert shared column** or **Create** → **Append shared column**. Select the shared column from the displayed list, as shown in Figure 4-10. Select the **Use Formula Only** option if you want to override the shared column display properties, such as font and column alignment, in the view. To lock the display properties, do not select this option. The inserted/appended column’s Programmatic Name (in the Properties → Advanced tab) is not inherited from the shared column, regardless of the setting of the Use Formula Only setting; therefore, its value can always be modified. However, the column value (formula) cannot be modified.



Select the shared column to insert. If you would like to customize the appearance of the shared column, please check the Use Formula Only checkbox

*Figure 4-10 Insert Shared Column window*

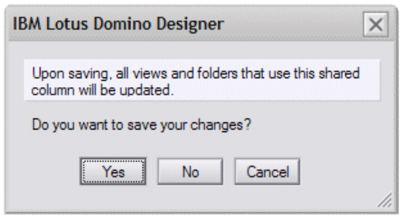
In the ITSO Electronics application, to use the Sales Person shared column in the "Sales People By Number" view, we open the view in Domino Designer and insert the shared column using the steps outlined earlier. Figure 4-11 shows the updated view with the inserted shared column.



| # | Region | Phone Number | Sales Person |
|---|---|---|---|
| 111555 | South East | 415-555-0987 | Lisa Chase |
| 111777 | East Coast | 617-555-4567 | Frank Grapengiesser |
| 111888 | South East | 617-555-8888 | Dick Larsson |
| 123111 | Midwest | 617-555-6789 | John Doe |
| 123222 | East Coast | 712-555-4444 | John Bergland |
| 123333 | West Coast | 415-555-1234 | Manny Santana |
| 123888 | Midwest | 617-555-7777 | Phil Monson |

*Figure 4-11 Updated Sales People By Number view with the inserted shared column Sales Person*

To edit the design of a shared column, open the **Shared Code** → **Columns** design list, and find the shared column in this list. A shared column can also be edited by selecting it in a view and selecting **Design** → **Edit Shared Column**. The user is prompted prior to saving changes that changes to a shared column definition are automatically reflected in views where the shared column is used, as shown in Figure 4-12. These updated views are resaved using the current user ID, and where **Use Formula Only** is selected, only the formula is updated.


IBM Lotus Domino Designer

Upon saving, all views and folders that use this shared column will be updated.

Do you want to save your changes?

*Figure 4-12 Saving a shared column*

To delete shared columns, select the shared column design element, select **Edit** → **Delete**, or right-click and select **Delete**, as shown in Figure 4-13. When a shared column is deleted, the shared column is changed to an unshared column in all views containing the column.

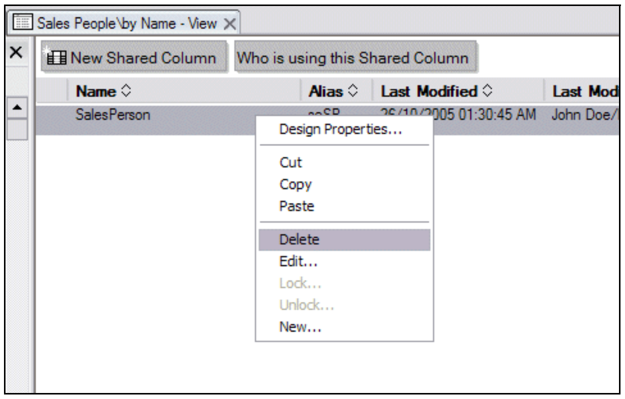
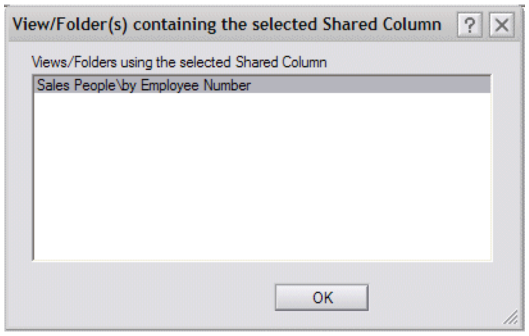


*Figure 4-13   Deleting a shared column*

By clicking the **Who is using this Shared Column** action button in the Columns design list, you can see at a glance which views and folders in the database are using the selected shared column. In the ITSO Electronics application, when we open the database in Domino Designer, select **Shared Code** → **Columns**, select the **SalesPerson** shared column from the list, and click the **Who is using this Shared Column** action button, this displays the information shown in Figure 4-14.


View/Folder(s) containing the selected Shared Column

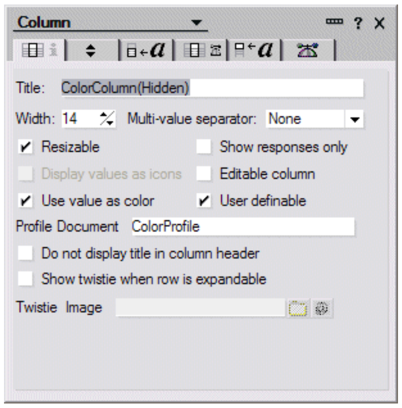
Views/Folders using the selected Shared Column

Sales People\by Employee Number

*Figure 4-14   Who is using this Shared Column message window*

## 4.4 Enhanced runtime customizations for views and folders

A column property called user definable enables developers to design a column formula based on profile documents in the database. In previous versions of Lotus Domino Designer software, use of this feature was limited to color coding a column. Lotus Domino Designer 7 provides increased profile document support in view and folder columns to permit more runtime options in applications. A good example of this is in the Lotus Notes 7 mail template. The Inbox folder contains a mail attention indicator column and a hidden column that controls color coding. Contents and behavior of both of these columns are defined by the user's mail preferences. See Figure 4-15 on page 88.



Title: ColorColumn(Hidden)
Width: 14 Multi-value separator: None
✓ Resizable
Show responses only
Display values as icons
Editable column
✓ Use value as color
✓ User definable
Profile Document ColorProfile
Do not display title in column header
Show twistie when row is expandable
Twistie Image

*Figure 4-15 User-definable Column option*

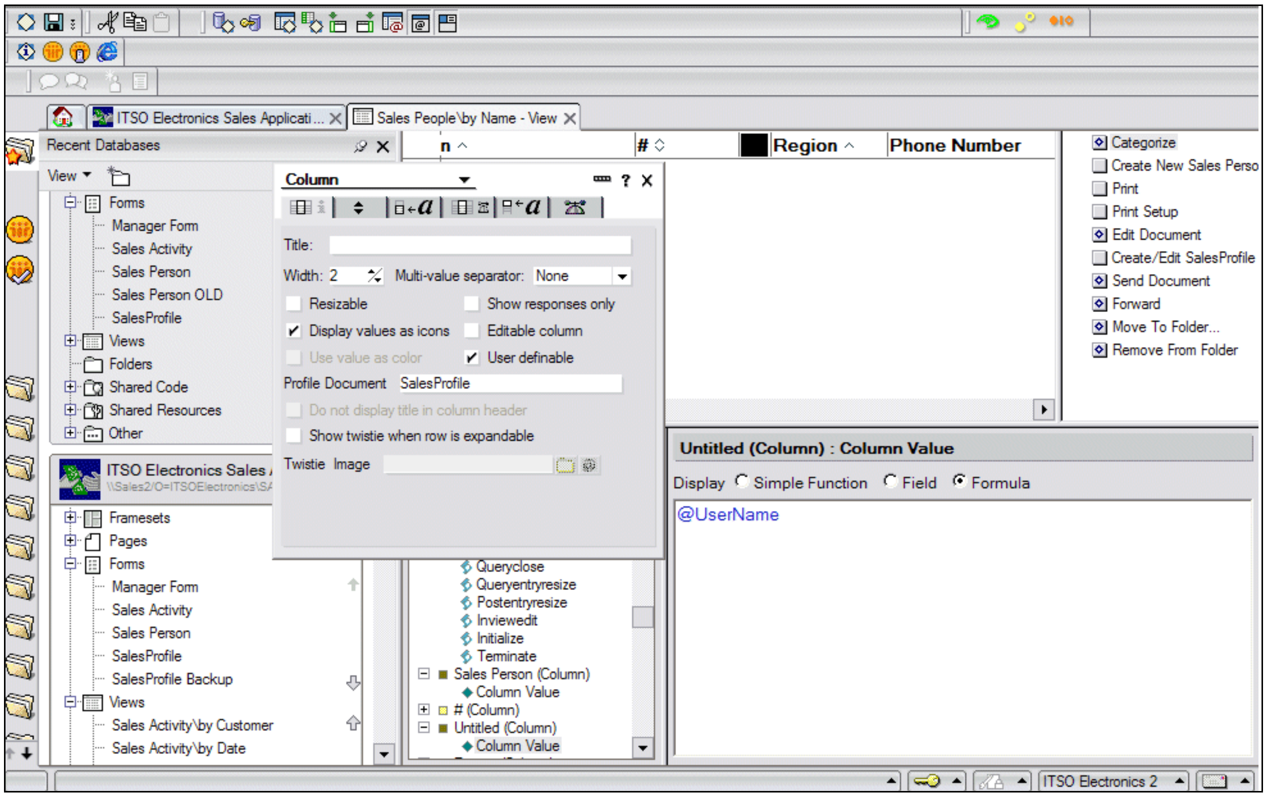
We want to enhance the ITSO Electronics application by adding a column that indicates the region that is highlighted for the month and also that indicates whether a document refers to the current user. A star icon is displayed if the region is the highlighted region for the month. A blue arrow indicates that the document refers to the current user. Otherwise, nothing is displayed in the column. Prior to adding a user profile column, we assume that the application has an existing profile document called SalesProfile. Only a single “shared” profile document can be used for user profile columns; therefore, when the profile document is created, the optional uniquekey parameter is not used. For more information about profile documents, refer to the IBM developerWorks® article *The hidden power of Profile documents*, available at:

`http://www.ibm.com/developerworks/lotus/library/ls-Profile_documents/`

To add the new user profile column in the “Sales Person By Name” view in our ITSO application:

1. Open the view in Domino Designer and insert a new column in the view by clicking **Create → Insert new column**.
2. Open the Column properties window and ensure that the following values are defined, as shown in Figure 4-16 on page 89:
   - Select **Display values as icon** because the user profile column will display an icon that corresponds to the highlighted region for the month or an indicator referring to the current user.
   - Select **User definable**.
   - Define the existing profile document in the database, `SalesProfile`, in the Profile Document field.
   - Use `@UserName` is used as the Column Value formula.

**Note:** The column value formula must be a non-constant formula, such as @Username, @Created, or @Random. Any non-constant formula can be used as the column value formula because the value is not evaluated. If a constant formula, such as “”, or a special text function, such as @DocNumber, are used, the user profile column will not display properly and a Replication or Save Conflict message will be displayed for each document in the view.



Column

Title:

Width: 2 Multi-value separator: None

- [ ] Resizable
- [ ] Show responses only
- [✓] Display values as icons
- [ ] Editable column
- [ ] Use value as color
- [✓] User definable

Profile Document SalesProfile

- [ ] Do not display title in column header
- [ ] Show twistie when row is expandable

Twistie Image

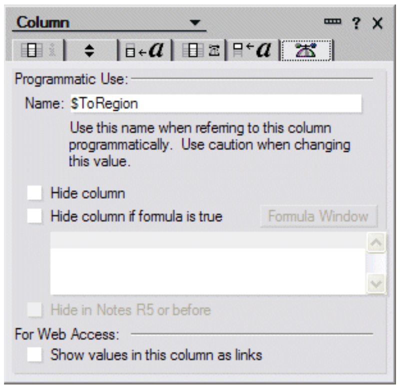
Untitled (Column) : Column Value

Display ( ) Simple Function ( ) Field (●) Formula

```
@UserName
```

Figure 4-16   Defining a user profile column

– On the Advanced tab, as shown in Figure 4-17, the Name field contains a field name that is in the profile document, in this case, `$ToRegion`. A formula is stored in this profile document field, which gets evaluated when a view row is displayed. In other words, the field *value* of the $ToRegion field is formula code whose output is the value to be displayed in the column. The *value formula* of $ToRegion is a formula whose output is that formula.



Column

Programmatic Use:

Name: $ToRegion

Use this name when referring to this column programmatically. Use caution when changing this value.

- [ ] Hide column
- [ ] Hide column if formula is true

Formula Window

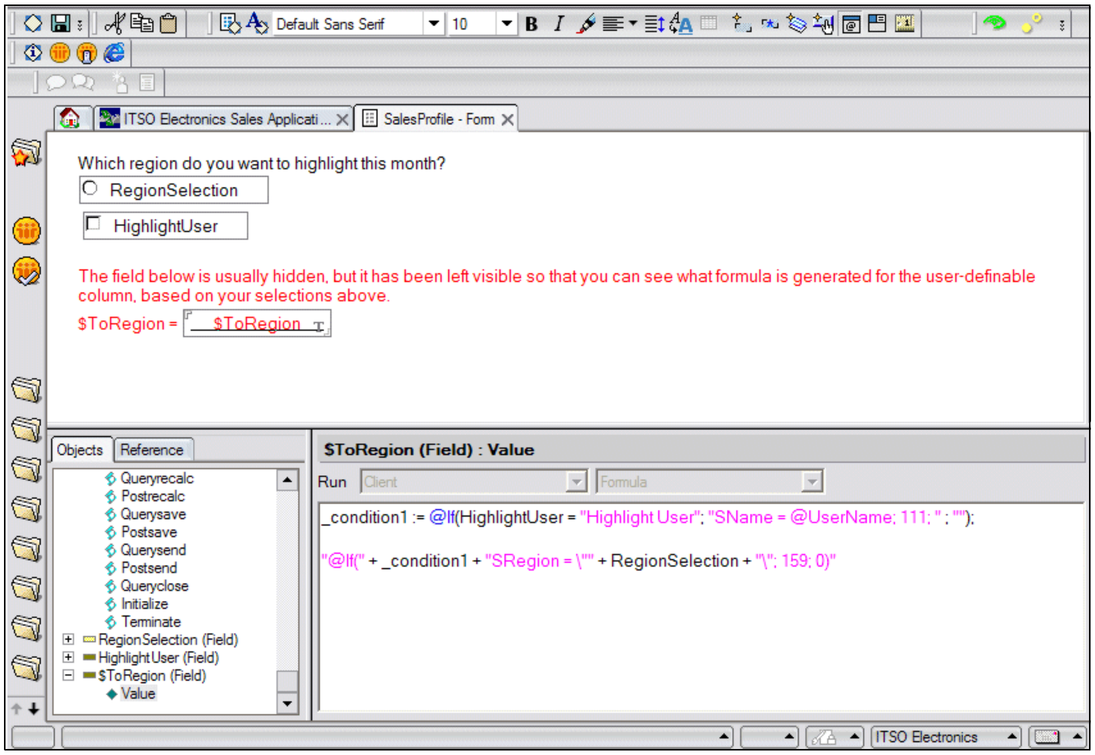
- [ ] Hide in Notes R5 or before

For Web Access:

- [ ] Show values in this column as links

Figure 4-17   Advanced tab in Column properties

The end user who edits the profile document, in most cases, would not be able to write this formula. A computed field that contains the formula, which in our example is $ToRegion, is used to calculate for the end user what the formula should be. This computed field formula must generate the formula that is stored in the field. Figure 4-18 shows design of the SalesProfile form and shows the field value in the $ToRegion field that generates the value formula.



Which region do you want to highlight this month?

RegionSelection

HighlightUser

The field below is usually hidden, but it has been left visible so that you can see what formula is generated for the user-definable column, based on your selections above.

$ToRegion = $ToRegion

**$ToRegion (Field) : Value**

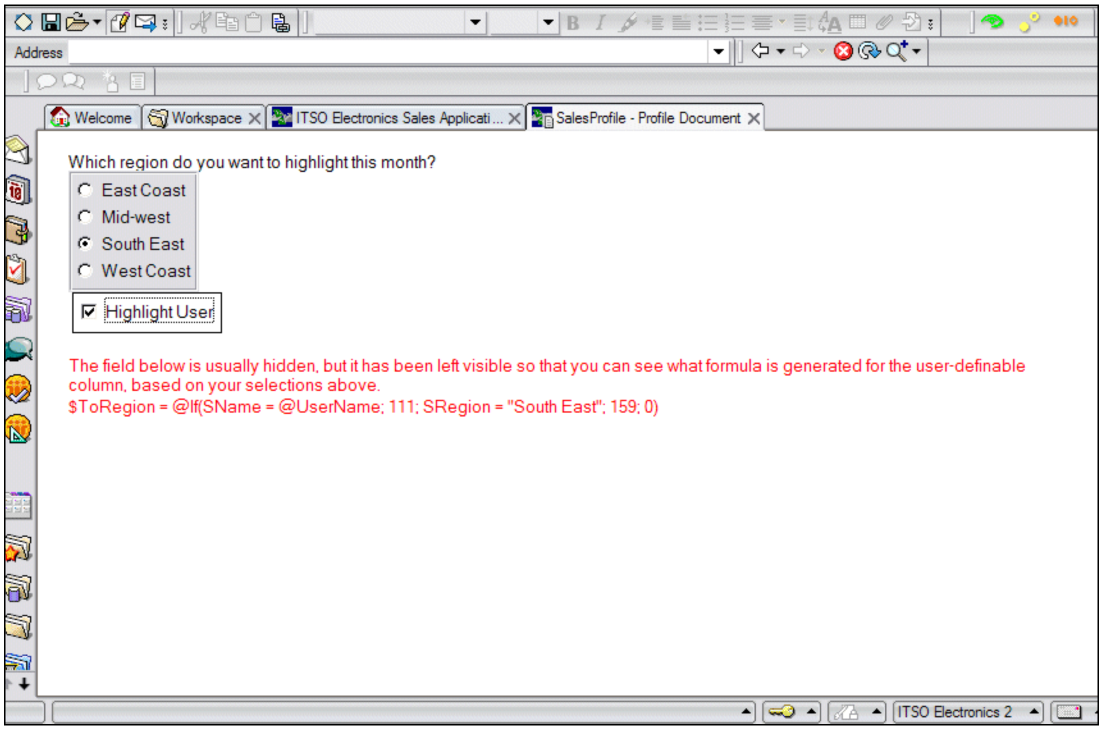
Run: Client, Formula

```
_condition1 := @If(HighlightUser = "Highlight User"; "SName = @UserName; 111; " ; "");

"@If(" + _condition1 + "SRegion = \"" + RegionSelection + "\"; 159; 0)"
```

*Figure 4-18 SalesProfile Form in Domino Designer*

3. Figure 4-19 shows the SalesProfile profile document. Enter values in the RegionSelection and HighlightUser fields. The formula of the $ToRegion field calculates a new formula that will be used for the user-definable column. For example, if the region selection is **South East** and **Highlight User** is selected, the value of $ToRegion will be:

```
@If(SName = @Username; 111; SRegion = “South East”; 159; 0)
```


Address
Welcome
Workspace
ITSO Electronics Sales Applicati ...
SalesProfile - Profile Document
ITSO Electronics 2

Which region do you want to highlight this month?

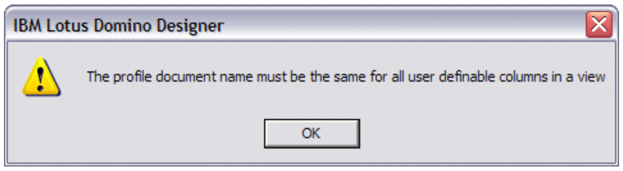
East Coast [✓ South East] Mid-west West Coast

[✓ Highlight User]

The field below is usually hidden, but it has been left visible so that you can see what formula is generated for the user-definable column, based on your selections above.
$ToRegion = @If(SName = @UserName; 111; SRegion = "South East"; 159; 0)

*Figure 4-19 SalesProfile Profile Document*

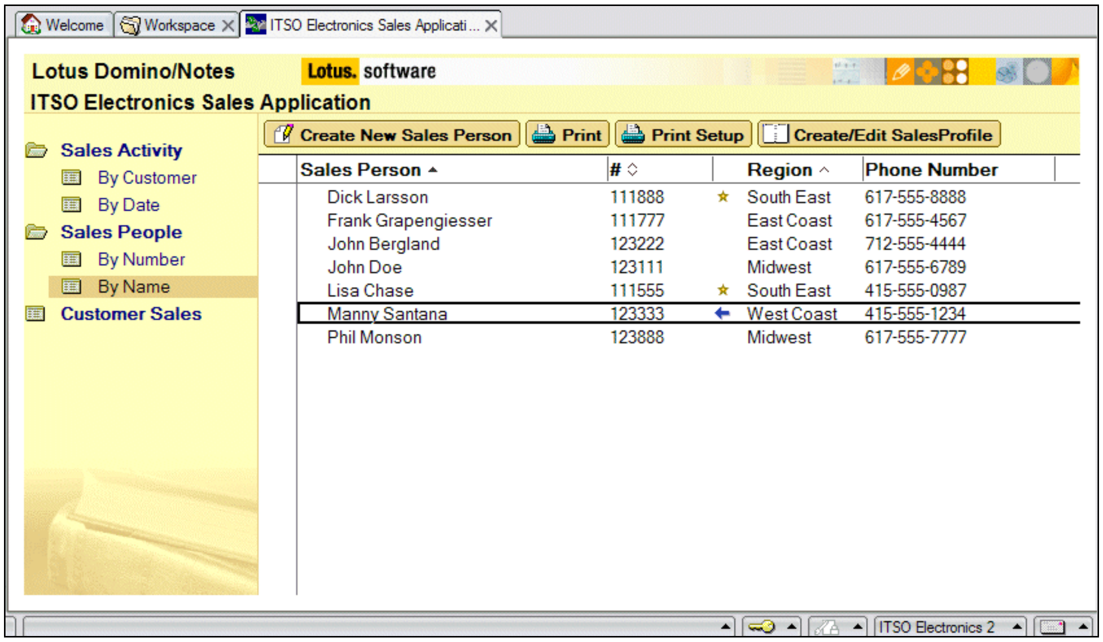
The User definable option will always be available, not just when Use value as color is selected. This enables using a profile document column for an icon or any other type of column. If a profile document is specified for multiple columns within the same view, it needs to be the same profile document for all columns. Otherwise, an error message is displayed when the view is saved, as shown in Figure 4-20.



The profile document name must be the same for all user definable columns in a view

*Figure 4-20 User profile column error message*

After the view is saved, the user profile column is displayed in the Sales People By Name view, as shown in Figure 4-21.



| Sales Person | # | | Region | Phone Number |
|---|---|---|---|---|
| Dick Larsson | 111888 | ☆ | South East | 617-555-8888 |
| Frank Grapengiesser | 111777 | | East Coast | 617-555-4567 |
| John Bergland | 123222 | | East Coast | 712-555-4444 |
| John Doe | 123111 | | Midwest | 617-555-6789 |
| Lisa Chase | 111555 | ☆ | South East | 415-555-0987 |
| Manny Santana | 123333 | ← | West Coast | 415-555-1234 |
| Phil Monson | 123888 | | Midwest | 617-555-7777 |

*Figure 4-21 User profile column displaying icons in view*

**Note:** Take care with quotation marks and backslashes in the computed field formula. Because many macro formulas contain quotation marks, and some contain backslashes, the computed value formula of the profile field must often contain quotation marks (or backslashes) inside of quotation marks, a common cause of confusion. This is one reason it is a good idea to make the computed field visible, as we have done here, just during development so that you can edit the fields, look at the resulting formula, and see whether it is syntactically correct. Also, when you insert data from other fields into your formula, keep in mind that it might also contain quotation marks or backslashes, so you might need to use @ReplaceSubstring(fieldname; "\\":"\""; "\\\\"; "\\\"") to convert them into proper string literals.

## 4.5 Additions to formula language

The Lotus Notes formula language has been updated in Domino Designer 7. New @Functions have been introduced in support of IBM DB2 and IBM Workplace Client Technology™ integration, as well as in support of security features of policy and ECL locking. A new **`@Command`** to locate every folder in which the selected document resides was also added.

The new commands and functions include:

- **`@AdminECLIsLocked`**: A new function that checks the current status of the Administration ECL in the name and address book.

- **@DB2Schema**: A new function that, given the name of a database as a text string, returns a text string containing the DB2 schema of that database if it is a db2nsf database or the empty string if it is not a db2nsf database. This function is useful in the SQL Query formula for Query Views, as discussed in 2.3, “Query Views” on page 18, where you must specify a formula to generate the SQL that retrieves the records you want. The SQL must use the fully qualified table name, including the schema name, and it is desirable to use the **@DB2Schema** function and not hardcode the schema name so that the design is portable to other servers and applications.
- **@IsDB2**: A new function that, given a server and file name or server and replica ID, indicates whether the specified database is backed by DB2 or not.
- **@IsEmbeddedInsideWCT**: A new function that indicates whether any part of the current Notes session is embedded inside of Workplace client.
- **@PolicyIsFieldLocked**: A new function that indicates whether a field is locked by an administration policy and cannot be modified.
- **@Command([DiscoverFolders])**: A new command that displays the “Folders containing current document” window.

**Note:** The **@IfError** function is obsolete in Lotus Domino Designer 7.0, but is supported for backward compatibility.

For more information about these new functions and commands, refer to the Lotus Domino Designer 7 Help, available at:

`http://www.lotus.com/ldd/doc/domino_notes/7.0/help7_designer.nsf`

## 4.6 Additions to LotusScript, COM, and OLE classes

LotusScript has been updated with improved capabilities, including:

- Support for the new code profiling feature, as discussed in 5.1, “Code profiling in the ITSO Electronics application” on page 96
- Improved document collection capabilities
- Expanded XML support
- Support for integration with Workplace Client Technology

One highlight of Lotus Domino Designer 7 is a new property that has been added to LotusScript. The property NotesUIDocument.ModifiedSinceSaved detects changes made to a document in edit mode since the last save.

For more information about the updates to LotusScript in Lotus Domino Designer 7, refer to the Lotus Domino Designer 7 Help, available at:

`http://www.lotus.com/ldd/doc/domino_notes/7.0/help7_designer.nsf`

## 4.7 Additions to Java and CORBA classes

Lotus Domino Designer 7 software supports a new version of Java, Java Virtual Machine (JVM™) 1.4.2, and a new version of the Extensible Markup Language (XML) parser. With a new check-box feature and Java debugging preferences, you can set up Java agents, Web services, and script libraries for remote debugging in the Lotus Notes client.

Lotus Domino Designer 7 adds the following features:

- Sun™ core Java documentation for Java and CORBA classes
- Support for the Sun Java™ 2 Platform, Technology Edition, V1.4.2
- The new AdministrationProcess.approveHostedOrgStorageDeletion method
- The new Database.getModifiedDocuments method
- The new DocumentCollection.UntilTime property
- Support for Java Platform Debugger Architecture (JPDA)

For more information about Java and CORBA support in Lotus Domino Designer 7, refer to the Lotus Domino Designer 7 Help, available at:

`http://www.lotus.com/ldd/doc/domino_notes/7.0/help7_designer.nsf`

5

# Diagnosing and troubleshooting the ITSO Electronics application

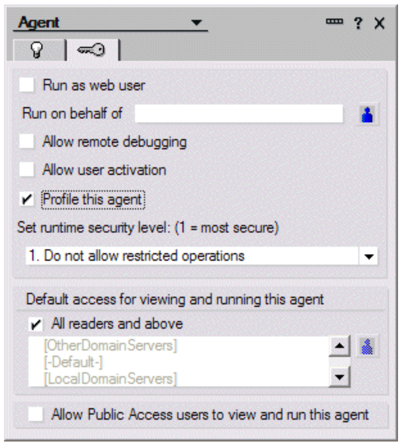
Lotus Domino Designer 7 introduces enhanced application diagnosis and troubleshooting capabilities with the addition of code profiling, advanced LotusScript debugging, and remote Java debugging. With the addition of these debugging tools, you can improve the performance of your Lotus Domino applications. Using the ITSO Electronics application as an example, we examine the following components of the Lotus Domino Designer debugging toolkit:

- Code profiling: Optimizing performance in the ITSO Electronics application
- LotusScript debugging: Leveraging enhanced LotusScript debugging elements in the ITSO Electronics application
- Java debugging: Remote Java debugging in the ITSO Electronics application

## 5.1 Code profiling in the ITSO Electronics application

Lotus Domino Designer 7 enables powerful code optimization with the addition of Profiler, a code profiling tool that enables the Lotus Domino developer to measure the time it takes to execute methods in agents, Web services, and script libraries in Lotus Notes and Domino applications. Measuring the performance of methods within the Lotus Domino application helps developers identify inefficiencies within the code and focus on optimizing those portions of the code that take the longest time to execute, and which therefore have the greatest performance impact. In this section, we examine how code profiling can be used to maximize the efficiency of the ITSO Electronics application.

Profiler examines and measures methods in both Java and LotusScript, including operations on Lotus Domino objects such as a Lotus Domino database. The developer can enable profiling in an agent or Web service by going to the Security tab of the properties window and selecting **Profile this agent** or **Profile this web service**. Profiling occurs each time an enabled agent or Web service runs. Agent profiling occurs on both the Lotus Notes client and Lotus Domino server. After turning on the profiling option in the second tab of the Agent properties window, as shown in Figure 5-1, the next time the agent runs, it will be profiled.



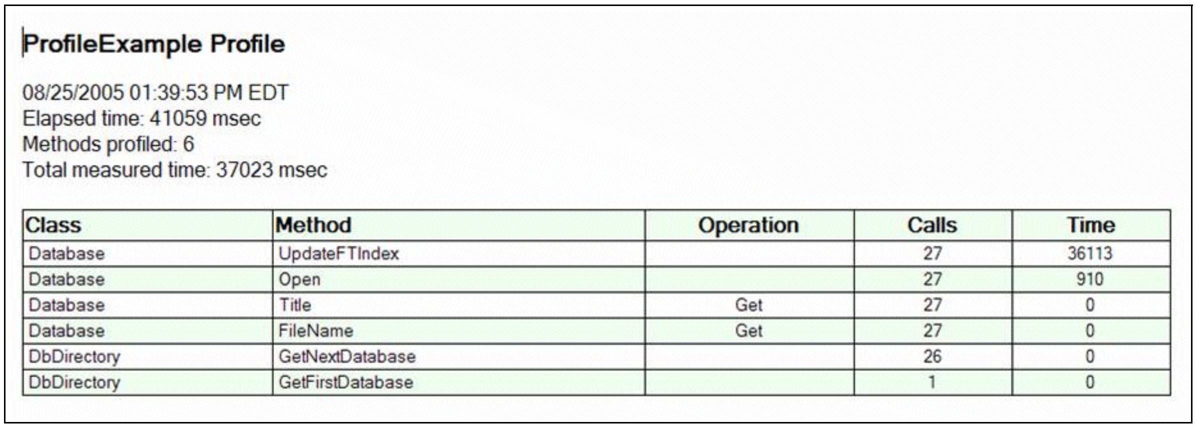
Run as web user
Run on behalf of
Allow remote debugging
Allow user activation
✓ Profile this agent
Set runtime security level: (1 = most secure)
1. Do not allow restricted operations
Default access for viewing and running this agent
✓ All readers and above
[OtherDomainServers]
[-Default-]
[LocalDomainServers]
Allow Public Access users to view and run this agent

*Figure 5-1 Agent properties*

Agents can be profiled regardless of whether they run as a scheduled agent, as a Web agent, or manually from the Action menu. The profiling information is stored in a Profile document in the database associated with the agent. To view profiling information, select the agent you are profiling in Lotus Domino Designer, and then select **Agent** → **View Profile Results**. At the top of each Profile document, the name of the agent and the time the profiling was done are recorded. Elapsed time is the total amount of time the agent ran, followed by a total measured time, which is typically somewhat smaller because time values are rounded down for display purposes. For example, the values under one millisecond are displayed as zeros in the Profile document. The profiling table contains a row for each Domino Objects method called, displayed in five columns:

- Class: The name of a Domino Objects class using normalized names such as Session, Database, and Document.
- Method: The name of a Domino Objects method or property using normalized names such as CurrentDatabase, AppendItemValue, and Save.
- Operation: for properties, the type of operation, either Get or Set.
- Calls: The number of times the method or property was called.
- Time: The amount of time the calls consumed in milliseconds. The symbol "<" means not enough time to calculate.

The information in the profiling table is sorted in descending order, showing the methods where the most amount of time was spent at the top. In this example, as shown in Figure 5-2, the profile output shows that the UpdateFTIndex method is the most time-consuming part of this agent.



**ProfileExample Profile**

08/25/2005 01:39:53 PM EDT
Elapsed time: 41059 msec
Methods profiled: 6
Total measured time: 37023 msec

| Class | Method | Operation | Calls | Time |
|---|---|---|---|---|
| Database | UpdateFTIndex | | 27 | 36113 |
| Database | Open | | 27 | 910 |
| Database | Title | Get | 27 | 0 |
| Database | FileName | Get | 27 | 0 |
| DbDirectory | GetNextDatabase | | 26 | 0 |
| DbDirectory | GetFirstDatabase | | 1 | 0 |

*Figure 5-2 Code profiling results*

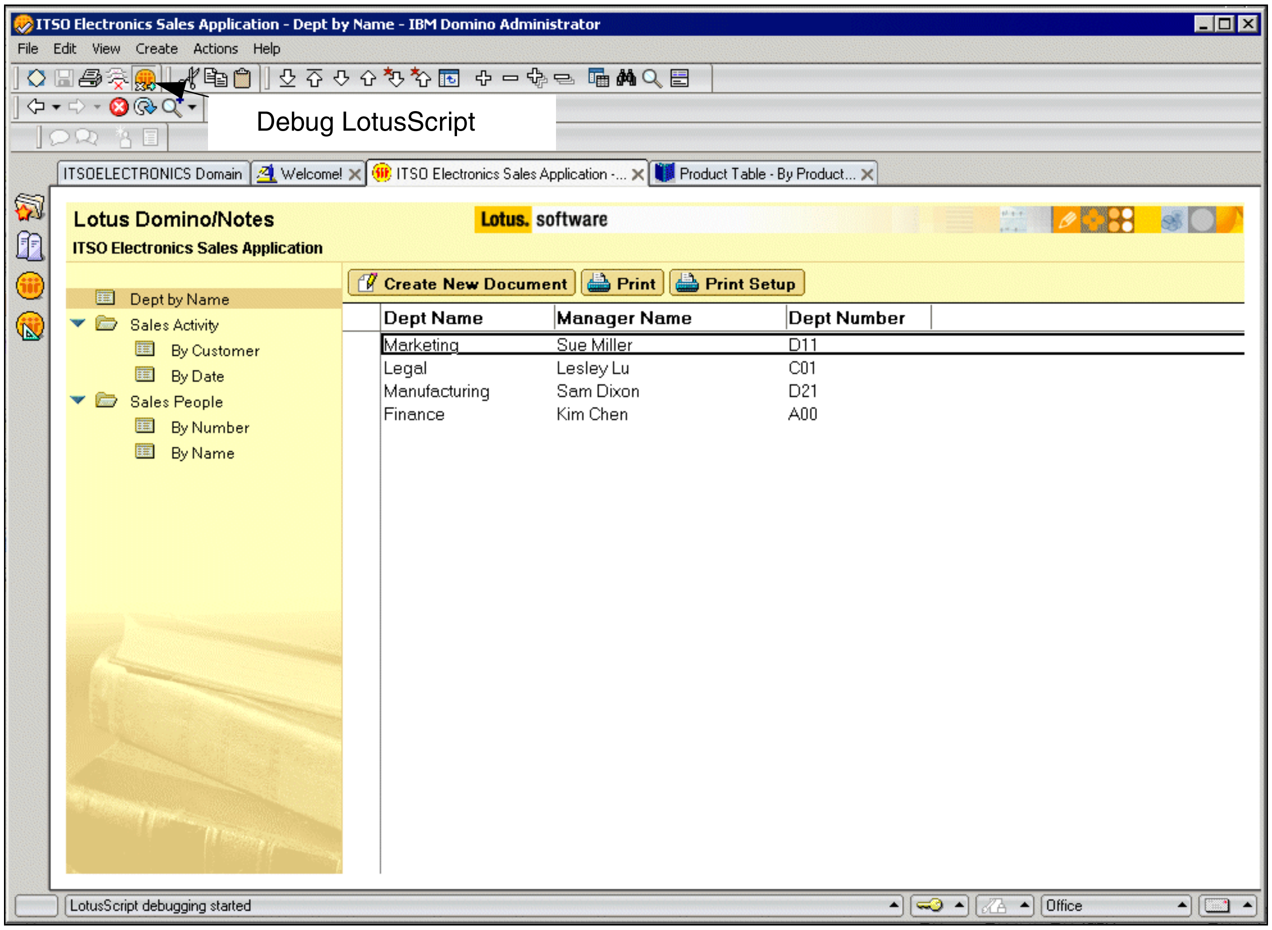
By capturing data on real-world execution of agents, code profiling provides developers with valuable information, enabling them to identify bottlenecks in code execution. Using this information, the developer can modify agent logic to improve execution time, thereby increasing user satisfaction and productivity.

## 5.2 LotusScript debugging in the ITSO Electronics application

Lotus Domino Designer 7 has strengthened LotusScript debugging capabilities by enabling a more intuitive use of this feature. Developers can now start and terminate debugging with an icon, in addition to initiating debugging through the menu system. The message indicating LotusScript debugging has been started or terminated goes to the status bar rather than a message window. In this section, we demonstrate the new LotusScript diagnostic features when debugging the ITSO Electronics application.

Follow these steps to debug LotusScript in the ITSO Electronics application using the new debugging capabilities:

1. To enable the debugger, click the **Debug LotusScript** icon on the toolbar. The message “LotusScript debugging started” appears on the status line, as shown in Figure 5-3 on page 98.

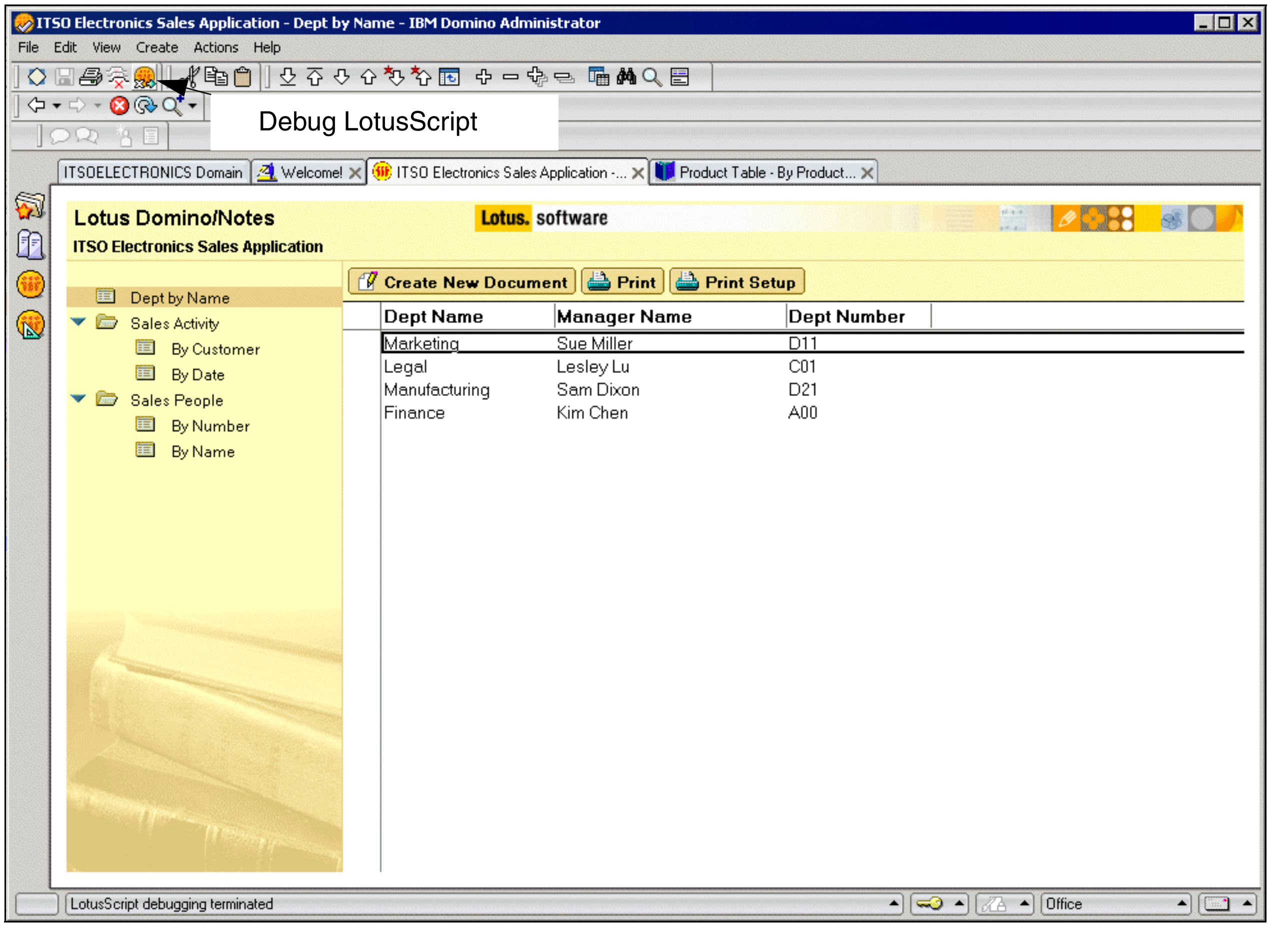

ITSO Electronics Sales Application - Dept by Name - IBM Domino Administrator
File Edit View Create Actions Help
Debug LotusScript
ITSOELECTRONICS Domain
Welcome!
ITSO Electronics Sales Application - ...
Product Table - By Product...
Lotus Domino/Notes
ITSO Electronics Sales Application
Lotus. software
Create New Document
Print
Print Setup
Dept by Name
Sales Activity
By Customer
By Date
Sales People
By Number
By Name
Office

| Dept Name | Manager Name | Dept Number |
|---|---|---|
| Marketing | Sue Miller | D11 |
| Legal | Lesley Lu | C01 |
| Manufacturing | Sam Dixon | D21 |
| Finance | Kim Chen | A00 |

LotusScript debugging started

*Figure 5-3 Starting the LotusScript debugger*

2. To disable the debugger, click the **Debug LotusScript** icon on the toolbar again. The message “LotusScript debugging terminated” appears on the status line, as shown in Figure 5-4 on page 99.


ITSO Electronics Sales Application - Dept by Name - IBM Domino Administrator
File Edit View Create Actions Help
Debug LotusScript
ITSOELECTRONICS Domain
Welcome!
ITSO Electronics Sales Application -...
Product Table - By Product...
Lotus Domino/Notes
ITSO Electronics Sales Application
Lotus. software
Create New Document
Print
Print Setup
Dept by Name
Sales Activity
By Customer
By Date
Sales People
By Number
By Name
LotusScript debugging terminated
Office

| Dept Name | Manager Name | Dept Number |
|---|---|---|
| Marketing | Sue Miller | D11 |
| Legal | Lesley Lu | C01 |
| Manufacturing | Sam Dixon | D21 |
| Finance | Kim Chen | A00 |

*Figure 5-4 Stopping the LotusScript debugger*

## 5.3 Java debugging in the ITSO Electronics application

Perhaps the most exciting addition to the Lotus Domino Designer 7 debugging toolkit is remote Java debugging. The Java remote debugger enables the Lotus Domino application developer to debug Java agents, Web previews, and script libraries running under control of a Lotus Notes client Java Virtual Machine (JVM) with a debugger that supports the Java Platform Debugger Architecture (JPDA), such as Rational Application Developer, based on Eclipse technology. Although you cannot debug Java code running on a Lotus Domino server, the JPDA offers a wealth of functionality for debugging Java code locally. In this section, we demonstrate remote Java debugging in the ITSO Electronics application using Rational Application Developer.

### 5.3.1 Enabling Java debugging on a Lotus Notes client

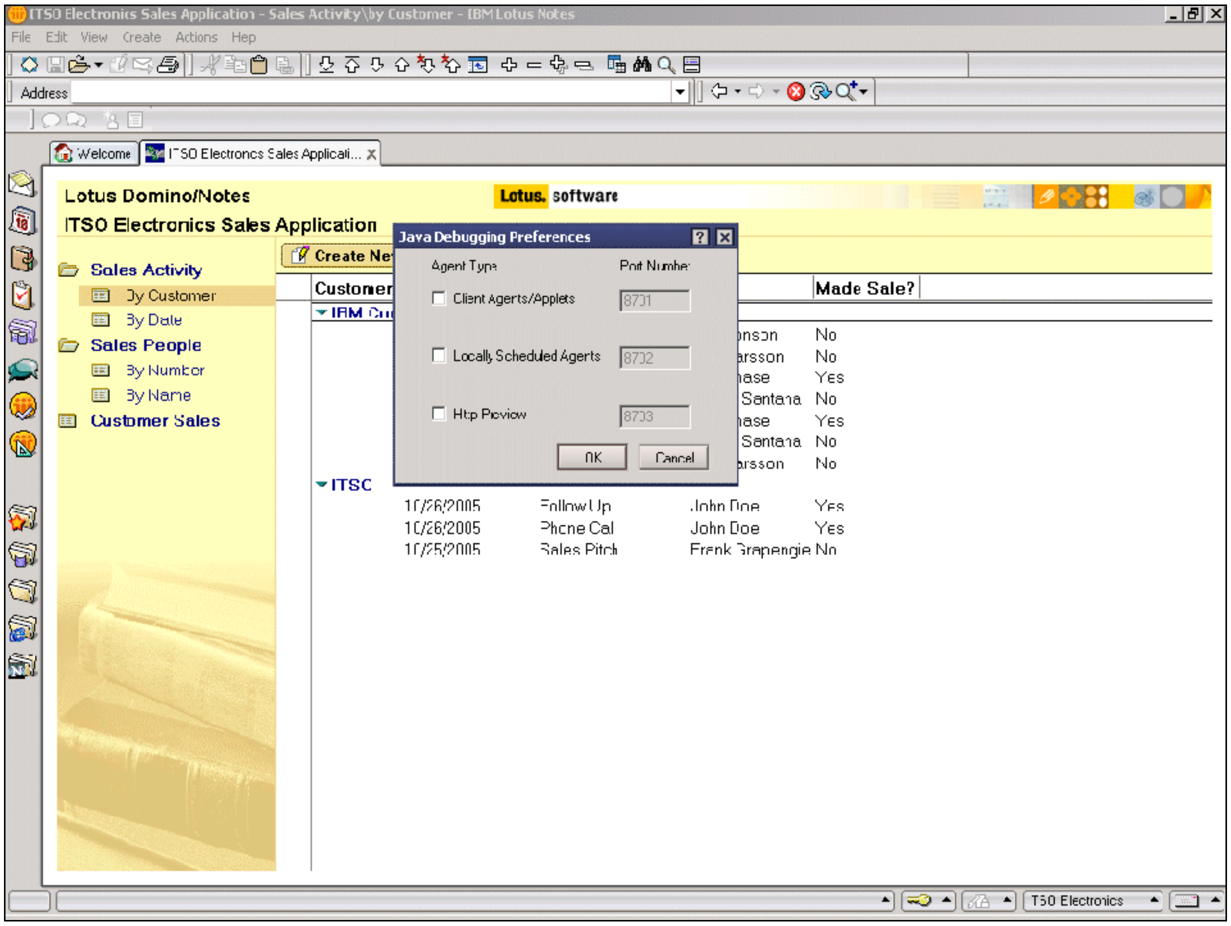
The Lotus Notes client supports Java debugging in the following contexts. Each context has its own JVM. Java code from a script library runs in the context of the calling code. Only one user can debug at a time in each context.

- Foreground: Java code that runs in the Lotus Notes client interactively, for example, an agent triggered from the Actions menu

- Background: Java code that runs in the Lotus Notes client under control of the task loader, for example, a locally scheduled agent
- Web preview: Java code being previewed in a browser through Lotus Domino Designer, for example, an applet on a form

Follow these steps to enable Java debugging in the ITSO Electronics application on a Lotus Notes client:

1. Select **File** → **Tools** → **Java Debugging Preferences**. This opens the Java Debugging Preferences window, as shown in Figure 5-5.


ITSO Electronics Sales Application - Sales Activity\by Customer - IBM Lotus Notes
File Edit View Create Actions Help
Address
Welcome
ITSO Electronics Sales Applicati...
Lotus Domino/Notes
ITSO Electronics Sales Application
Lotus. software
Create New
Sales Activity
By Customer
By Date
Sales People
By Number
By Name
Customer Sales
Customer
Made Sale?
IBM Cu
ITSC
ITSO Electronics

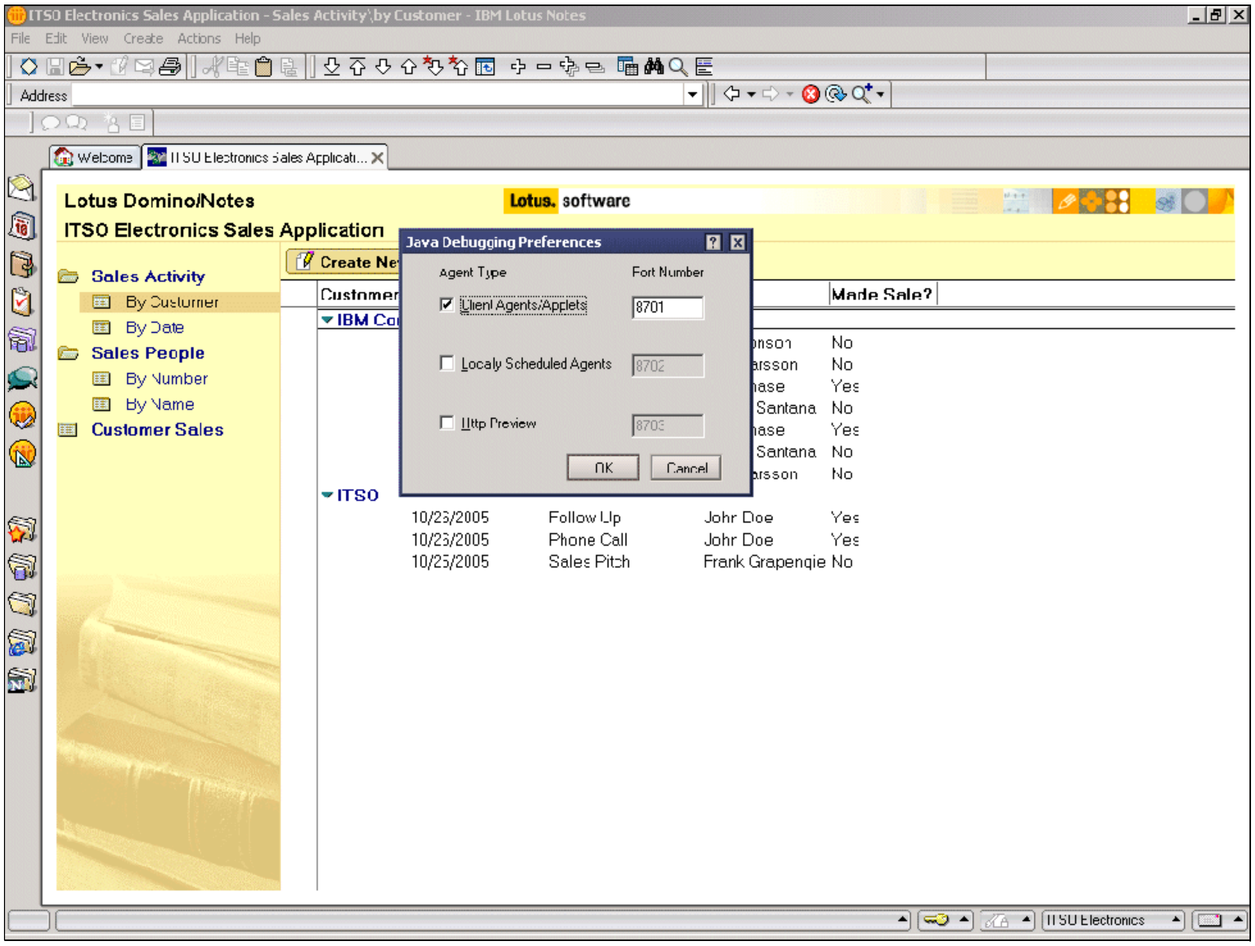
Java Debugging Preferences

| Agent Type | Port Number |
|---|---|
| ☐ Client Agents/Applets | 8701 |
| ☐ Locally Scheduled Agents | 8702 |
| ☐ Http Preview | 8703 |

OK Cancel

Figure 5-5 Java Debugging Preferences window

2. To enable foreground debugging, select **Client Agents/Applets** and specify a port number to connect the Lotus Notes and debugger computers, as shown in Figure 5-6 on page 101. Clear this to disable it.


ITSO Electronics Sales Application - Sales Activity\by Customer - IBM Lotus Notes
File Edit View Create Actions Help
Address
Welcome
ITSO Electronics Sales Applicati...
Lotus Domino/Notes
ITSO Electronics Sales Application
Lotus. software
Sales Activity
By Customer
By Date
Sales People
By Number
By Name
Customer Sales
Create New
Customer
Made Sale?
IBM Co
ITSO
ITSO Electronics

Java Debugging Preferences

| Agent Type | Port Number |
|---|---|
| ☑ Client Agents/Applets | 8701 |
| ☐ Localy Scheduled Agents | 8702 |
| ☐ Http Preview | 8703 |

OK Cancel

| | | | |
|---|---|---|---|
| | | onson | No |
| | | arsson | No |
| | | hase | Yes |
| | | Santana | No |
| | | hase | Yes |
| | | Santana | No |
| | | arsson | No |
| 10/23/2005 | Follow Up | John Doe | Yes |
| 10/23/2005 | Phone Call | John Doe | Yes |
| 10/23/2005 | Sales Pitch | Frank Grapengie | No |

*Figure 5-6 Java Debugging Preferences with Client Agents/Applets selected*

3. To enable Web preview debugging, select **Http Preview** and specify a port number to connect the computer running the Lotus Notes client. Clear this to disable it.

   Pick a port number that is free. To view the ports in use on a Windows machine, issue this command at a command line:

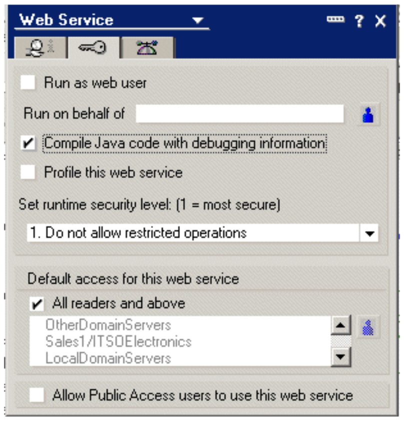
   ```
   > netstat -a
   ```

   Java debugging is disabled by default.

If changes are made to the foreground or background preference, Lotus Notes must be restarted. If changes are made to the Web preview preference, the preview must be restarted.

## 5.3.2 Enabling Java debugging in an agent, Web service, or script library

To enable Java debugging, go to the Security tab of the Agent or Web Service properties window, or to the Script Library properties windows, and select **Compile Java code with debugging information**, as shown in Figure 5-7 on page 102.


Web Service

[ ] Run as web user

Run on behalf of

[✓] Compile Java code with debugging information

[ ] Profile this web service

Set runtime security level: (1 = most secure)

1. Do not allow restricted operations

Default access for this web service

[✓] All readers and above

OtherDomainServers
Sales1/ITSOElectronics
LocalDomainServers

[ ] Allow Public Access users to use this web service

*Figure 5-7 Compile Java code with debugging*

When compiling Java code to be debugged outside Lotus Domino Designer, compile it with the **-g** option. After modifying an agent or Web service, export the source code to a file that is accessible to your debugger.

For information about JPDA, see:

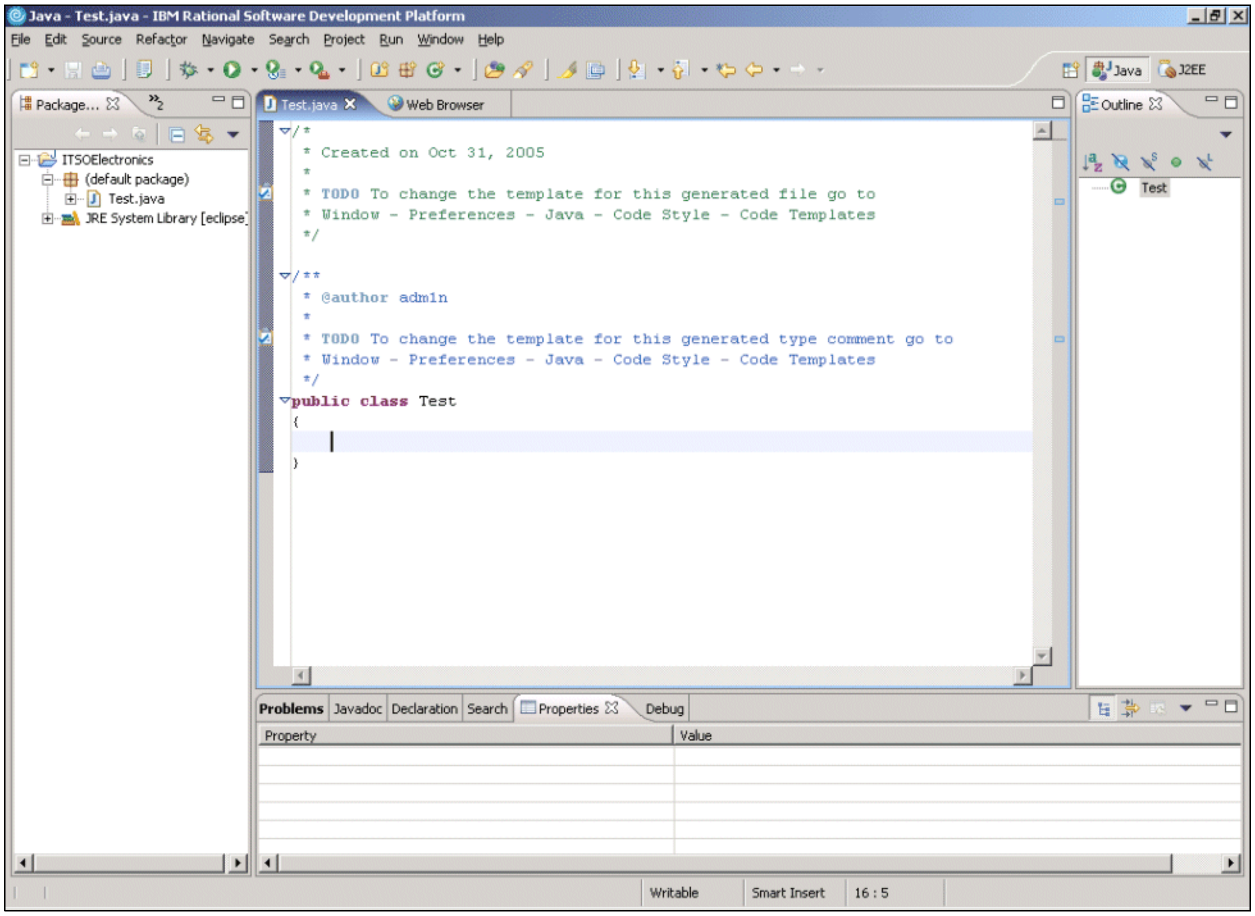
http://java.sun.com/products/jpda

## 5.3.3 Connecting a debugger to the JVM

The following instructions assume you are using the Rational Application Developer tool, based on Eclipse technology. You have to interpolate for other debuggers. Perform the following steps:

1. Start the Lotus Notes client and open the ITSO Electronics application containing the agent or Web service to be debugged with Java debugging enabled.
2. Start the debugger.

3. Create a Java project and switch to the Java perspective, as shown in Figure 5-8. Select **File** → **New** → **Project**. Select **Java** in the Project window, enter a name for the project, and click **Finish**. Switch to the Java perspective. Alternatively, open the project if it already exists.


Java - Test.java - IBM Rational Software Development Platform
File Edit Source Refactor Navigate Search Project Run Window Help
Java
J2EE
Package...
ITSOElectronics
(default package)
Test.java
JRE System Library [eclipse]
Test.java
Web Browser
Outline
Test
Problems Javadoc Declaration Search Properties Debug
Property
Value
Writable
Smart Insert
16 : 5

```
/*
 * Created on Oct 31, 2005
 *
 * TODO To change the template for this generated file go to
 * Window - Preferences - Java - Code Style - Code Templates
 */

/**
 * @author admin
 *
 * TODO To change the template for this generated type comment go to
 * Window - Preferences - Java - Code Style - Code Templates
 */
public class Test
{

}
```

*Figure 5-8 Viewing the Java perspective*

4. To import the source Java files, right-click the project folder in the left pane and select **Import**, as shown in Figure 5-9.

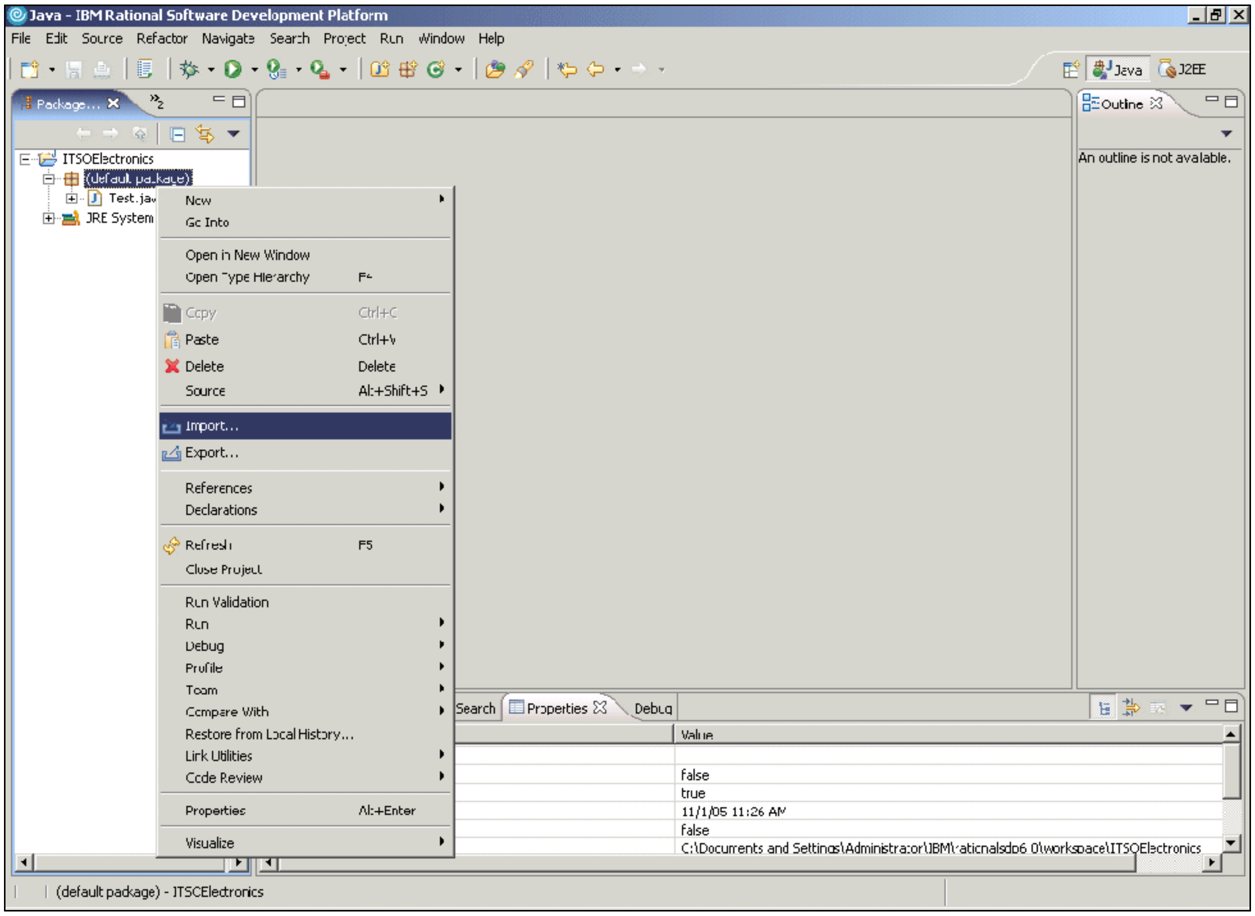
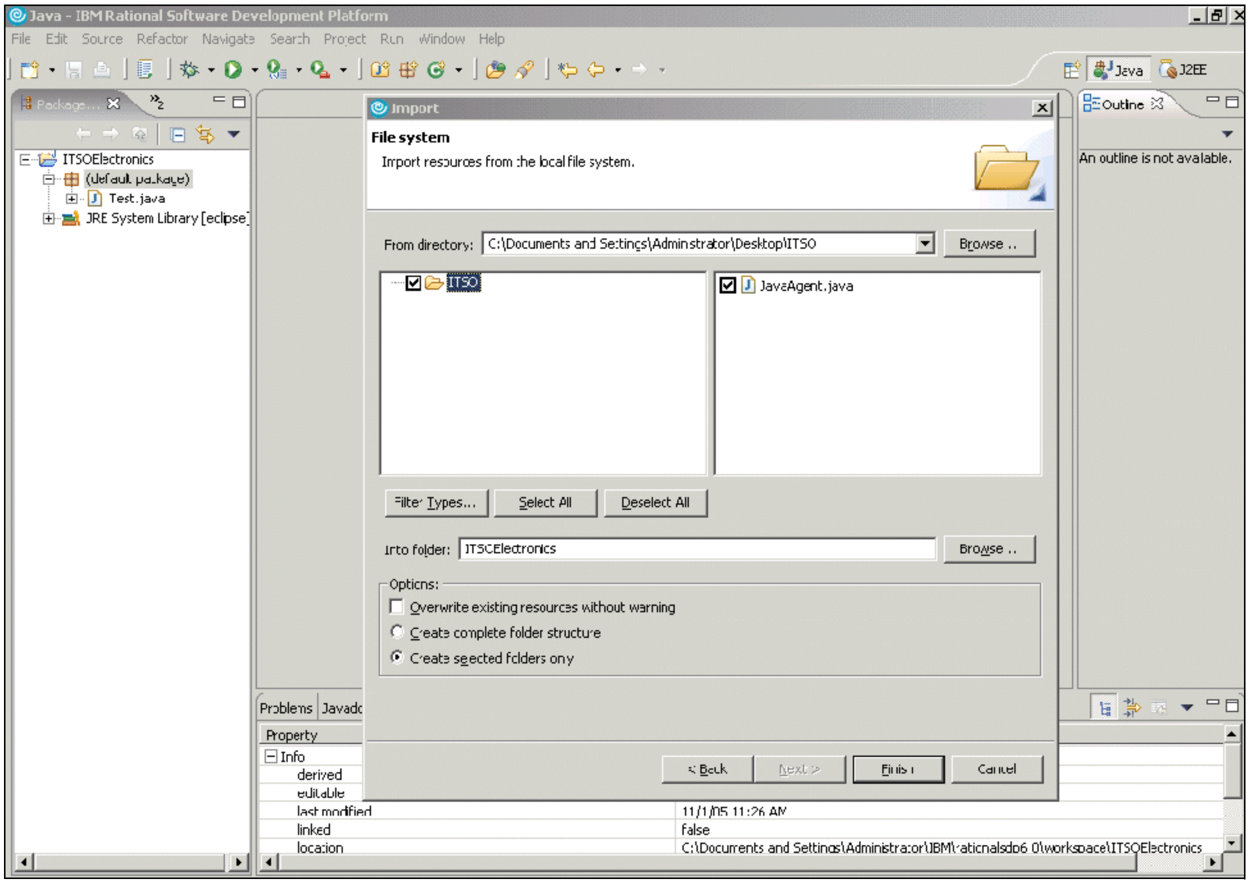


Figure 5-9 Import into the Java perspective

5. Select the type of files to import, and then browse to and select the Java source files. Select the folder to store the imported files, as shown in Figure 5-10.



File system
Import resources from the local file system.

From directory: C:\Documents and Settings\Administrator\Desktop\ITSO

[✓] ITSO
[✓] JavaAgent.java

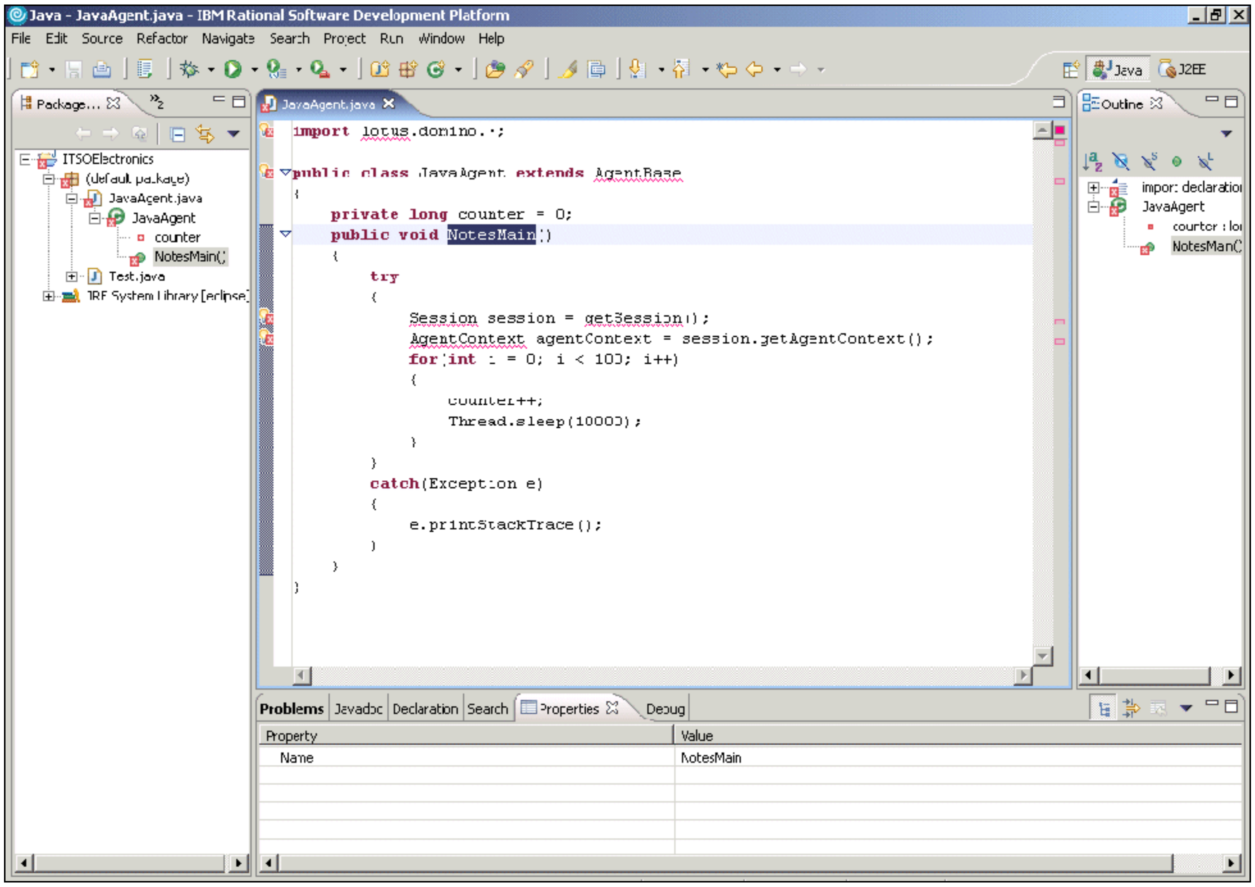
Into folder: ITSOElectronics

Options:
[ ] Overwrite existing resources without warning
( ) Create complete folder structure
(●) Create selected folders only

| | |
|---|---|
| derived | |
| editable | |
| last modified | 11/1/05 11:26 AM |
| linked | false |
| location | C:\Documents and Settings\Administrator\IBM\rationalsdp6.0\workspace\ITSOElectronics |

Figure 5-10 Java perspective and the Import window

6. Click **Finish** to import the files. Only source code that is imported is available to the debugger. If the Java source code is not imported, you can see the execution thread and can access variables, but you cannot see the source code and set breakpoints. See Figure 5-11.



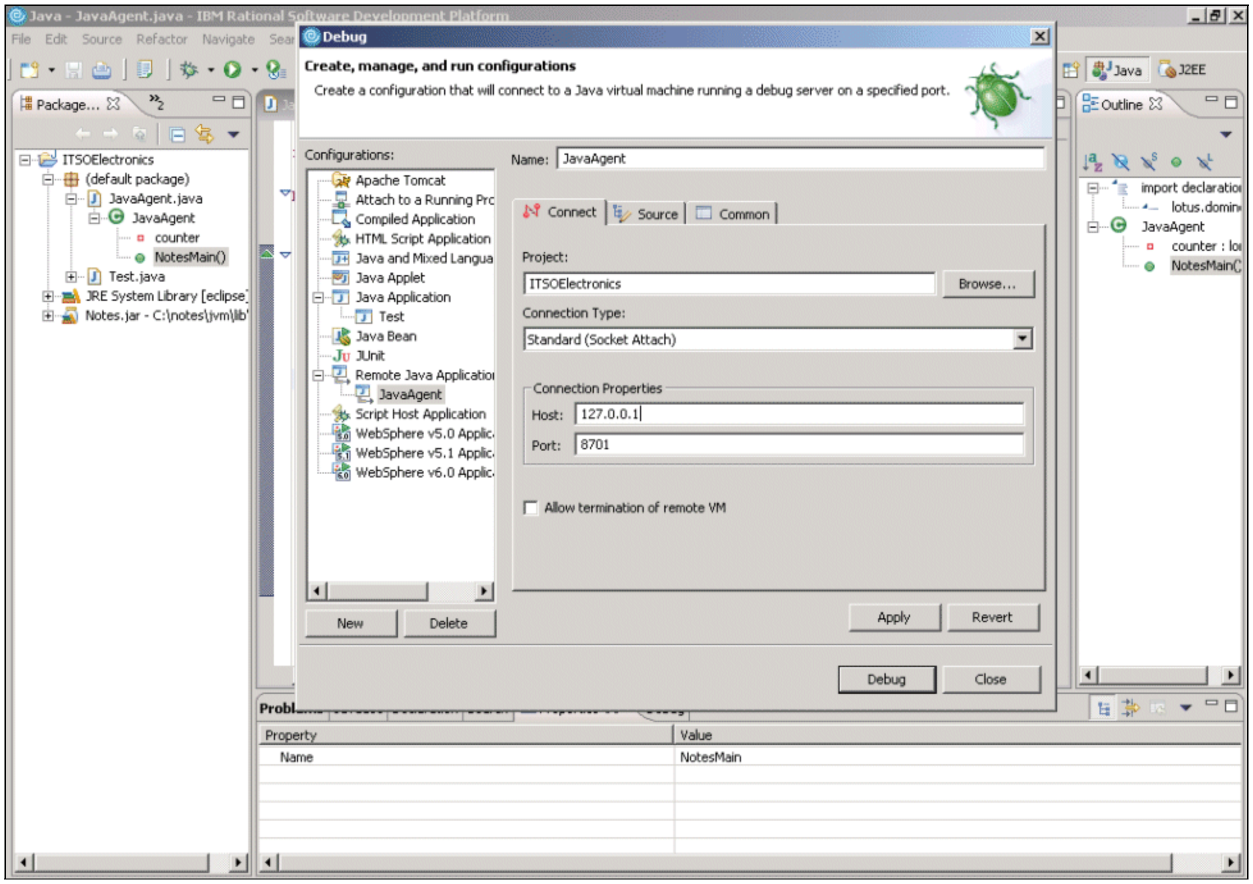
```
import lotus.domino.*;

public class JavaAgent extends AgentBase
{
    private long counter = 0;
    public void NotesMain()
    {
        try
        {
            Session session = getSession();
            AgentContext agentContext = session.getAgentContext();
            for(int i = 0; i < 100; i++)
            {
                counter++;
                Thread.sleep(10000);
            }
        }
        catch(Exception e)
        {
            e.printStackTrace();
        }
    }
}
```

| Property | Value |
|---|---|
| Name | NotesMain |

*Figure 5-11 Imported Java source code*

7. In Lotus Notes, start the agent or Web service to be debugged.

8. In Rational Application Developer, select **Run** → **Debug**, and enter a name for the debug configuration, as shown in Figure 5-12. Go to the Connect tab. Select the project, enter the host name or address of the Lotus Notes computer, and enter the Java debug port number on the Lotus Notes client. The host address is `127.0.0.1` if the Lotus Notes client and Eclipse are on the same machine. Click **Apply** (only necessary the first time or if you change the configuration), and then click **Debug** to attach to the debugger.



Create, manage, and run configurations

Create a configuration that will connect to a Java virtual machine running a debug server on a specified port.

Name: JavaAgent

Project: ITSOElectronics

Connection Type: Standard (Socket Attach)

Host: 127.0.0.1

Port: 8701

☐ Allow termination of remote VM

| Property | Value |
|---|---|
| Name | NotesMain |

*Figure 5-12 Setting the connection parameters for debugging*

After you attach the debugger, you will see the execution threads for the JVM. An agent's thread looks something like this:

```
Thread(AgentThread: JavaAgent)(Running)
```

You can suspend the agent's thread to gain control over it. At this point, the usual debugger features become available (setting breakpoints, stepping, continuing, viewing variable values, and so on).

The Java agent must run long enough for you to attach the debugger to the JVM. You might delay execution at the beginning of the procedure, for example, by inserting a loop with sleep statements.

If Lotus Notes and the debugger are on separate computers, and you get the error “Failed to connect to remote VM,” the designated port is probably taken by another process. Try specifying another port number and restarting Lotus Notes.

For detailed information about remote Java debugging, refer to the Lotus Domino Designer 7 Help, available at:

`http://www.lotus.com/ldd/doc/domino_notes/7.0/help7_designer.nsf`

## 5.3.4 Security considerations

Java debugging over a network is insecure. You should restart Lotus Notes or the Web preview without Java debugging enabled when you are not debugging Java code. To disable Java debugging, see 5.3.1, “Enabling Java debugging on a Lotus Notes client” on page 99.

A

# Additional material

This Redpaper refers to additional material that can be downloaded from the Internet as described here.

## Locating the Web material

The Web material associated with this Redpaper is available in softcopy on the Internet from the IBM Redbooks Web server. Point your Web browser to:

`ftp://www.redbooks.ibm.com/redbooks/REDP4102`

Alternatively, you can go to the IBM Redbooks Web site at:

**ibm.com**/redbooks

Select the **Additional materials** and open the directory that corresponds with the Redpaper form number, REDP4102.

## Using the Web material

The additional Web material that accompanies this Redpaper includes the ITSO Electronics Databases.zip file.

### How to use the Web material

Create a subdirectory (folder) on your workstation, and unzip the contents of the Web material ZIP file into the folder. Then, copy the Domino databases into your Notes/Domino data directory, for example, c:\notes\data, on your local workstation or on your Lotus Domino server.

# Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this Redpaper.

## IBM Redbooks

For information about ordering these publications, see "How to get IBM Redbooks" on page 112. Note that some of the documents referenced here may be available in softcopy only.

- *Domino Designer 6: A Developer's Handbook*, SG24-6854
- *Lotus Domino 6.5.1 and Extended Products Integration Guide*, SG24-6357
- *Using the Domino JSP Custom Tags: An Approach to Portalizing Domino Applications*, REDP-3902
- *IBM Lotus Domino Application Portlet: Configuration and Tips*, REDP-3917
- *Security Considerations in Notes and Domino 7*, REDP-4104

## Online resources

These Web sites and URLs are also relevant as further information sources:

- IBM developerWorks: Lotus

  `http://www.ibm.com/developerworks/lotus`
- Lotus Domino Administrator 7 Help

  `http://www.lotus.com/ldd/doc/domino_notes/7.0/help7_admin.nsf/`
- IBM Lotus documentation site

  `http://www.lotus.com/ldd/doc`
- IBM training and certification site

  `http://www.ibm.com/software/sw-lotus/services/education.nsf/wdocs/educationhomepage`
- DB2 Information Center

  `http://publib.boulder.ibm.com/infocenter/db2help/index.jsp`
- Lotus Domino 7 Designer Help

  `http://www.lotus.com/ldd/doc/domino_notes/7.0/help7_designer.nsf`
- IBM developerWorks: *All about AutoSave in Lotus Notes/Domino 7*

  `http://www.ibm.com/developerworks/lotus/library/autosave/`
- IBM developerWorks: *Lotus Notes/Domino 7 Web Services*

  `http://www.ibm.com/developerworks/lotus/library/nd7-webservices/`
- IBM developerWorks: *Which style of WSDL should I use?*

  `http://www.ibm.com/developerworks/webservices/library/ws-whichwsdl/`
- IBM developerWorks: *The Hidden Power of Profile Documents*

  `http://www.ibm.com/developerworks/lotus/library/ls-Profile_documents/`

- Developing Applications with IBM Lotus Domino Enabled for DB2, E-learning Edition: Self Paced course, N7D400SP

  http://www.ibm.com/software/sw-lotus/services/education.nsf/81F04813DC9CE3C7852566DA004C5CB2/A6DF60493B0D2B13852570A7006C7E9B

- Introducing IBM DB2 Concepts course, N7D020

  https://education.lotus.com/rw/lewwschd.nsf/5FDC13DD71D1143185256575001E5133/E55E8003DCEC00FA8525701A006959C4

- Data types descriptions

  http://www.w3.org/TR/xmlschema-2

- *Web Service Architecture* guide

  http://www.w3.org/TR/2004/NOTE-ws-arch-20040211

- Apache Axis

  http://ws.apache.org/axis/

- Apache SOAP

  http://ws.apache.org/soap/

- Java Platform Debugger Architecture (JPDA)

  http://java.sun.com/products/jpda

## How to get IBM Redbooks

You can search for, view, or download Redbooks, Redpapers, Hints and Tips, draft publications and Additional materials, as well as order hardcopy Redbooks or CD-ROMs, at this Web site:

**ibm.com**/redbooks

## Help from IBM

IBM Support and downloads

**ibm.com**/support

IBM Global Services

**ibm.com**/services

# Lotus Domino 7 Application Development

Redpaper

**Add storage and relational capabilities with DB2 integration**

**Extend and expose Domino applications with Web services**

**Usability improvements and new design and diagnostic tools**

With Version 7, IBM Lotus Domino Designer software builds on its reputation for being a premier collaborative application development tool for Lotus Domino software-based applications. As an integral part of the IBM Workplace family, Lotus Domino Designer software helps companies improve employee productivity and build and deploy Lotus Domino applications more quickly, thereby enabling organizations to be more responsive to changing business initiatives. New features focus on tighter integration with Web standards, more interoperability with IBM technologies and ease of use.

This IBM Redpaper shows application developers how to use the powerful, new features in Domino 7 by enhancing an example application.