Create a deployment topology diagram in IBM Rational Software Architect

Donald Bell

October 08, 2009

This article shows how to use IBM® Rational® Software Architect to create a deployment topology diagram. After completing this exercise, you will be able to model the physical deployment of either a Java™ Enterprise Edition (JEE™) or Microsoft® .NET Web application.

A deployment topology diagram shows how a solution's software and hardware will be deployed and configured. The diagram not only provides a great communication vehicle to share key deployment information among infrastructure team members, but it is also a great communication vehicle for the development and infrastructure teams to share information between groups.

This article shows how to use IBM® Rational® Software Architect to create a deployment topology diagram. The example is based on a fictitious Microsoft® .NET Web application rather than a Java™ Enterprise Edition (JEE) Web application, because the .NET illustration shows how to create and harvest reusable topology units. This would not be shown in a typical JEE Web application scenario. However, after completing the exercise in this article, you will be able to model the physical deployment of either a JEE or .NET Web application.

Prerequisite

The author of this article assumes that you have some familiarity with topology modeling. If you have no knowledge, we recommend that you watch the five-minute "Managing deployment architecture with the topology editor" video or complete the "Planning deployment with the topology editor tutorial," both of which are included with the Help system of IBM® Rational® Software Architect.

The author used IBM® Rational® Software Architect for WebSphere® Software, Version 7.5.4, to create the topology example, and the same instructions apply to Rational Software Architect Standard Edition Version 7.5.4. However, the steps and the screen captures will be slightly different with earlier releases of Version 7.5. Nonetheless, you should be able to translate the steps and screen images to an earlier version.
Background

This article shows how to model the fictitious JKE Furniture Catalog Web application. As background, the Furniture Catalog is a .NET Web application that uses an IBM® DB2® database backend. The deployment topology created in this example documents how this application is deployed in the production environment.

Normally, an application development team starts by creating logical models and then iterating through logical and physical models together. However, this article focuses only on creating a deployment topology. There are many legitimate reasons that you might create a deployment topology first, but explaining how to determine what level of model or topology to start with is beyond the scope of this article. Instead, this exercise covers the mechanics of how to create a deployment topology.

Create a physical topology

Rational Software Architect topology models need to be stored in a project. As a general rule, topologies are stored with the solution's UML models.

1. Create a topology by switching to the Deployment perspective and selecting from the main menu File > New > Topology.

Figure 1 contains the values used for the JKE Furniture Catalog's deployment topology. There are four fields and values that you need to complete:

- **Namespace**: com.jke.furniture.deployment
- **Type**: Deployment
- **Contract**: Make All Units Private By Default
- **Add topology to palette**: Unchecked
In general, the Namespace field uses the Java™ package construct. This topology is a deployment topology; therefore, the package name ends in `deployment`. This naming convention is not required, but is useful for consistency.

The Contract field is set to "Make All Units Private By Default," because this is the best practice to follow when creating a deployment topology. A deployment topology is the lowest-level topology, so by setting the Contract field to "Make All Units Private," Rational Software Architect displays all validation problems as errors rather than warnings.

**Note:**

Warnings are shown as errors because the warnings can be fulfilled or resolved by importing the topology into another topology, which the latter topology can addresses. However, if the unit is marked "private," then all problems must be fixed in the current topology, because private units cannot be accessed when they imported by another topology. Using the "Private" policy on the topology makes all of the units private by default, which turns warnings into errors.

2. Click **Finish**.
3. Read the "Welcome to the Topology Editor" quick start Help and then dismiss the Help text by clicking the X in the top-right corner.
You will then have an empty deployment topology that shows a single topology diagram with a name that is the same as the topology model.

**Note:**
On the disk, the diagram file is separate from the topology file. When you are using a version control system, you need to put both files into the version control system.

**Model a DB2 database**

The JKE Furniture Catalog application gets all of its information from a DB2 database named FURNDB. JKE has standardized on DB2 Version 8; therefore, the included topology unit templates can be used to model the deployment of the databases.

There are two ways to add a new unit to the topology diagram.

- You can select the unit template from the palette (the DB2 database template is in the palette's Configurations drawer, under the DB2 stack).
- Or you can use the CTRL+T keyboard shortcut.

Generally, the CTRL+T option is the quickest way, because you can type the name of the unit. This saves you the time of looking for the unit inside of the palette drawers.

1. To create the DB2 database, click on the newly created topology diagram and then press CTRL+T.
2. When the drop-down window appears, type *database (the asterisk is a wildcard search character).
3. As you type, the dialog will filter the units. Select the **DB2 Database** unit (see Figure 2.).

**Figure 2. CTRL+T drop-down menu**

4. After placing the DB2 unit on the topology diagram, rename the unit to FURNDB by hovering your mouse cursor over the unit and clicking the pencil icon. In the popup window, leave **Link with Database Name** checked.

**Tip:**
Generally, when renaming units, leave the Link with the XXXX attribute checked.

You now have a unit that represents the actual database that stores the data. If you are experienced in using DB2, you already know that the database needs to be hosted on an instance of DB2. Just in case you don’t know, Rational Software Architect informs you of this problem by reporting a hosting error. You can see this hosting error by hovering your cursor over the red X in the FURNDB unit’s lower-left corner.

**Tip:**
The Topology Status view, which is located in lower-center of the perspective by default, is a useful view, because it displays all of the diagram’s problems. This view helps you successfully model your topology by guiding you through fixing every error.

5. Create a DB2 8 Instance unit either by using the palette or the CTRL+T option, and name the unit FURNINST.
6. After renaming the unit to FURNINST, drag the FURNDDB unit onto the FURNINST unit.

The dragging action will result in hosting the FURNDDB unit on the FURNINST unit. The hosting fixes the FURNDDB unit’s problem, and the hosting relationship indicates that the database runs within the DB2 Instance.

Notice that the newly created FURNINST unit has an error saying that the DB2 Instance needs to be hosted in a DB2 System installation. Fixing this problem is easy:

7. Simply create a DB2 8 System unit by using the palette or CTRL+T and create the host relationship by dragging the FURNINST unit onto the newly created DB2 8 System unit.

After creating the DB2 8 System and creating the host for the FURNINST DB2 instance on the system, your topology diagram should look like Figure 3.

**Figure 3. A DB2 installation that is hosting the FURNDDB database**

Unlike the FURNDDB database, where its only problem was fixed when it was hosted on the FURNINST DB2 instance, the FURNINST DB2 instance still has unresolved problems. However, this is acceptable at the moment, because the problems will get resolved after the deployment topology is further documented with the DB2 installation hosted on its server.

**Host the database on a Linux server**

The previous section documented the topology of the DB2 8 installation that is hosting the FURNDDB database. This section documents the physical server that will host the DB2 installation
documented in the previous section. However, this section is going to document the server first and then make it so the server can host the DB2 installation.

**Document the server**

As context, JKE Furniture runs their DB2 installations on x86 Linux® pizza box servers with dual CPUs.

1. On the topology diagram, create a unit for the x86 server by using the **x86 Server unit** template (that template is in the palette’s Hardware drawer). JKE uses IBM® System x®, 3450 servers, so rename the x86 Server unit to **IBM x3450**.

**Note:**
The server unit is named after its hardware model instead of its hostname because the hostname will be specified in operating system unit.

By default, Rational Software Architect documents x86 servers as having a single CPU; therefore, the newly created unit needs to be updated to reflect the fact that JKE Furniture has this as a two-CPU machine. The number of CPUs is stored in an attribute on the unit’s x86 Server capabilities.

2. To update the attribute, select the **IBM x3450** unit on the topology diagram and then go to the Properties view.
3. In the Properties view select the **Capabilities** tab. Because the x86 Server unit has only the one capability named x86 Server, that capability’s attributes are displayed by default.
4. Set the CPU Count attribute’s value to 2.

Figure 4 highlights the sequence of the steps:

A. X86 Server > IBM x3450
B. Properties tab
C. Capabilities
D. X86 Server
E. CPU Count: [enter] 2

**Figure 4. Navigation path to update the server's CPU count**
5. By default, JKE Furniture always uses 4 GBs of memory in its database servers. So after updating the CPU Count attribute, scroll down the attribute's pane and update the Memory Size attribute to 4 GB.

Typically, most people want to see the CPU count and the memory size of the servers. Starting in V7.5.1 Rational Software Architect allows for the attributes to be displayed on the diagram.

6. To display these attributes on the topology diagram, select the IBM x3450 unit on the topology diagram and then select **Diagram > Unit Details.** from the menu.

7. In the Show Unit Presentation Details window, put a checkmark next to both the CPU Count and the Memory Size attributes as Figure 5 shows.

**Figure 5. Show Unit Presentation Details view**

8. After clicking **OK**, the diagram will show the attributes on the server unit that Figure 6 illustrates.
Now that you have documented the physical server, it is time to document the operating system that is installed on the server.

**Document the operating system**

JKE Furniture runs SUSE Linux.

1. Create a SUSE Linux Server 9 unit on the topology diagram (using the **CTRL+T** option will probably be the easiest).

2. Name the newly created unit. In the popup window, leave the **Link with Hostname:URI** field checked and, in the Hostname field, enter `furn-db.jke.com`. Leave the URI field blank.

**Note:**
The URI attribute is used to identify the URI location of the UNIX® server's disk drive; which we will assume is not important in this fictitious JKE Furniture scenario.

The newly created operating system unit needs to be hosted on a server.

3. Host the `furn-db.jke.com` operating system unit on the IBM System *x 3450* server by dragging the operating system onto the server.

The `furn-db.jke.com` server stack still shows an error, and hovering your cursor over the stack's red Xs reveals that the error is coming from the root user account unit. The root user account needs to have its password set. In JKE Furniture, only the system administrators know the root password; therefore, it is not important to document the root user account, and we will delete it from the topology.

**Note:**
Rather than deleting the root user unit, another option is to mark the root user account unit as publicly editable, which would change the error to a warning. Marking the unit as public-editable indicates that the unit values can be satisfied later through an importing topology.
4. Delete the root user account from the operating system by selecting it and then using **CTRL + DEL**. This keystroke sequence removes the root user from the topology model.

**Note:**
Simply using the **DEL** key would remove the unit from the diagram only, not from the topology model.

The server unit now has zero errors. However, looking at the Topology Status view reveals that there is a hosting unit error.

5. In the Topology Status view, expand the **Host Units** category.
6. Select the **Unit ("DB2 System") must be hosted (Operating System)** error. Notice that, when you do this, you get four options for quick fixes.
7. Fix the problem by double-clicking this one:
   **Host "DB2 System" on "furn-db.jke.com" on IBM x3450**

After Rational Software Architect fixes the problem, the diagram will look like Figure 7.

**Figure 7. DB2 system hosted on furn-db.jke.com**

The double-lined arrow in the diagram is an alternative notation that shows a unit being hosted on another unit.

The DB2 stack still has a couple of errors that need to be resolved.

1. Click on the FURNINST unit’s red X.
2. Select the **No admin users found in OS Host(FURNIST)** error, and double-click on the **Create DB2 Instance Admin User** quick fix.
3. In the Create DB2 Instance Admin User window, enter `db2furn` in the DB2 Instance Admin User ID field, and set the password to something simple, such as `abc123`.
4. Click **OK**, and notice that Rational Software Architect creates a user account in the operating system and draws a dependency link between the FURNINST DB2 instance and the newly created db2furn user account.
Moving on to the other errors, hovering over the DB2 System's red X it reveals that the `classpath` entries are set to a Windows path.

5. Update each `classpath` value with a prefix, such as `/opt/IBM/db2v8.2/SQLLIB/`, and use the quick fix for each error.

**Note:**
An alternative to fixing the problems through the errors and quick-fix dialog is to select the DB2 System unit and then update the `classpath` attributes on the unit's various capabilities in the Properties view. This is similar to what you did when you updated the CPU Count and Memory Size attributes of the physical server.

After resolving all of the `classpath` errors, the topology will have zero problems.

6. For consistency, drag the DB2 System unit onto the furn-db.jke.com unit.

This will update the diagram so that all of the units are displayed in a tree view. The tree structure helps keep the diagram clean while displaying lots of information. After dragging the DB2 system onto the furn-db.jke.com unit and resizing, the diagram will look like Figure 8.

**Figure 8. IBM System x 3450 server hosting a DB2 database on the Linux operating system**

Create units for the ASP.NET Web application

There will always be applications and technology domains for which Rational Software Architect does not provide topology templates. For example, Version 7.5.4 does not include topology modeling templates for Microsoft® .NET applications.

The JKE Furniture Catalog is a Microsoft ASP.NET application. Therefore, the topology units that needed to support modeling a .NET system must be created. This takes a little time, but Rational Software Architect makes it easy to reuse and share units that you make.

This section focuses on creating the new topology units required to handle the ASP.NET platform. A later section explains how to make the units reusable.
JKE Furniture hosts its ASP.NET applications on Microsoft® Windows® 2003 servers. This means that the JKE Furniture Catalog will be an ASP.NET 2.0-compliant application that runs in Microsoft® Internet Information Server (IIS) 6.0.

**Create the IIS 6.0 unit**

1. To create the IIS 6.0 unit, create a new Software Install unit on the topology diagram by using `CTRL+T` or the palette.
2. Edit the unit's name by clicking the pencil icon. In the popup window, check **Link with Product Name:Version String:Release:Major Version:Modifier**.
3. Set new popup values with the following values:
   - **Product Name**: Internet Information Server
   - **Version String**: 6.0
   - **Major Version**: 6

**Figure 9. Software Install popup window**

4. By default, the newly created unit is called "SoftwareInstallUnit." To change the unit's name, select the new unit and then select the **Diagram > Unit Style** menu item.
5. In the Unit Style window, update the value in the **Unit title** field to **Internet Information Server 6.0**, as Figure 10 shows, and then click **OK**.
Figure 10. Unit Style window

The unit now accurately reflects the software from the outside; meaning that, from a diagram's visual perspective, the unit looks correct. However, the power of using Rational Software Architect for typology modeling is that the units know what their requirements and capabilities are. Therefore, the next steps involve documenting the IIS 6.0 requirements and capabilities.

Document the IIS requirements and capabilities

Hovering your mouse cursor over the IIS 6.0 unit's red X reveals that the unit has a requirement to be hosted on an operating system. Although this requirement is correct in the generic sense, notice that there is a quick fix to host IIS 6.0 on the Linux server. This quick fix is not really valid, so the unit's hosting requirement should be refined to reflect that IIS needs to be hosted on a Windows O/S (this way, someone less experienced does not accidentally plan to host IIS on Linux).

1. Update the operating system requirement by selecting the IIS 6.0 unit and then, in the Properties view, select the Requirements tab.
2. Select the Host on an Operating System requirement and click the Type Selector button (the one with ellipses, or three periods, to the right of the Type attribute filed).
3. In the Set Requirement Type dialog, select the `os.WindowsOperatingSystem` type.

Figure 11 shows screen captures of the process and highlights the Type Selector button.

**Figure 11. Updating the Operating System requirement to be Windows**

4. After updating the requirement's Type attribute, update the Caption attribute to say "Host on a Windows Operating System".

Hovering your cursor over the unit's red X now shows the updated error message and the quick fix of hosting IIS 6.0 on the Linux server is not offered.

The IIS 6.0 requirements could be further refined to reflect supported versions of the operating system and not having another IIS instance installed on the same machine. However, this article won't go into those details. Just having the requirement of a Windows O/S is good enough for this purpose.

The next part of the IIS 6.0 unit that needs to be updated is the unit's capabilities. IIS 6.0 hosts ASP.NET 2.0 Web applications. To document that the JKE Furniture Catalog application runs within IIS, the IIS 6.0 unit needs to document its capability to host ASP.NET applications.

5. Assuming that the IIS 6.0 unit is still selected in the diagram, in the Properties view go to the **Capabilities** tab.

6. Add a new capability by using the Property view's **Add Capability** button and, in the **Add Capability** window, select the `core.Capability` as shown in Figure 12.
7. After creating the capability, set the capability's Caption to "ASP.NET Runtime," and select **hosting** from the Link type drop down field.

8. To make this capability uniquely identifiable so that other units can require the capability, click the **Stereotypes** sub-tab and add a Stereotype named **ASP.NET Runtime**.

Figure 13 shows the navigation steps required.

**Figure 12. Add a core capability**

![Add a core capability](image)

**Figure 13. Add a stereotype to a capability**

![Add a stereotype to a capability](image)

9. After adding the Stereotype, go back to the **Main** sub-tab and add one attribute by clicking on the Property view's **Customize Attributes** button.

10. Name the new attribute **version** as Figure 14 shows.

11. Be sure to leave the attributes' type as **String**.
Figure 14. The Customize Attributes window

After you click the Customize Attributes dialog’s **OK** button, the capability panel will display entry fields for the newly created attribute.

13. Set the Version attribute’s value to **2.0**.

After you have finished these updates, the ASP.NET runtime attributes should look like Figure 15.

Figure 15. ASP.NET runtime capabilities

Like the requirements, the capabilities could be further documented, but the capabilities are now documented enough to enable the documentation of the JKE Furniture Catalog application.

Create a unit for the JKE Web application

1. Create a new Software Install unit by using **CTRL+T** or the palette, and name it **JKE Furniture Catalog Web App** by using the pencil icon and linking the name to the product name as you did previously with IIS.

2. Using the same steps as for the Internet Information Server 6.0 unit, update the unit title to be **ASP.NET 2.0 Web Component**.

Your diagram should now look like Figure 16.
The JKE Web App's outside appearance is correct now. However, its requirements need to be updated to reflect the application's runtime requirements.

3. Select the **JKE Furniture Catalog Web App** in the diagram.
4. Then, in the Properties view, select the Requirements tab.

The unit's hosting requirement needs to be updated to reflect the ASP.NET runtime requirement.

5. Start by changing the "Host on an Operating System" requirement's caption to **Host on an ASP.NET Runtime**.
6. Also change the Type attribute to **core.Capability**.

Generally speaking, the Web application is now partially requiring the IIS 6.0 unit, because the requirements type matches the IIS 6.0 unit's capability type (core.Capability). However, the hosting requirement needs to be further refined by using constraints.

7. Staying on the "Host on an ASP.NET Runtime" requirement, click the **Constraints** sub-tab.
8. Add a Stereotype constraint by using the **Add Constraint** button.
9. Set new constraints values with the following values:
   - **Caption**: Requires ASP.NET Runtime
   - **Includes**: ASP.NET Runtime
10. Add a second constraint of an Equals constraint by using the **Add Constraint** button.
11. Set new constraints values with the following values:
    - **Attribute**: Version
    - **Caption**: .NET 2.0
    - **Value**: 2.0

These constraints are the same as the values for the IIS 6.0 unit. Figure 17 highlights the required mouse clicks and shows the constraints.
Figure 17. ASP.NET runtime constraints

You have now met the requirement to ensure that the ASP.NET Web component is hosted on an ISS server (unit).

**Document the database requirement**

The JKE application also uses data from the FURNDB database, so the next step is to document this requirement.

You could manually create the corresponding requirement and capability as you did for the ASP.NET runtime, but there is an easier way to create corresponding dependencies.

**Tip:**

You can configure Rational Software Architect to automatically create and match dependencies between a single requirement and capability in the Topology Editor section of Preferences. Just check the **Create a link automatically when only one match exists** field. It is recommended that you check this field to work around a known defect in Version 7.5.4.

1. In the Palette view, select the **Dependency Link** tool and draw a link from the **JKE Furniture Web App** unit to the **FURNDB** unit. Rational Software Architect will report that there is not a match, but after you release the mouse button, the Dependency Link dialog will still appear.
2. In the Source Unit(s) column, select the **JKE Furniture Catalog Web App** unit, and click the **Create Both** button.

Figure 18 shows the navigation path.
3. After creating the matching requirement and capability, select the newly created requirement on the left and click the **Edit Properties** button.
4. Rename the requirement to **Requires FURNDB**, and add a stereotype constraint for **FURNDB**.
5. Then rename the capability by selecting the capability on the right and clicking the **Edit Properties** button. The capability's Properties dialog will open in the same location as the requirements dialog. Rename the capability to **Provides FURNDB** and then click **OK** on the Create Dependency Link dialog.

After completing the dialog, the diagram will have errors (see Figure 19).

### Figure 19. Web application with dependency errors

6. Click on the dependency link's error and select this quick fix: **Add missing stereotypes to capability "Provides FURNDB"**

The requirements for the JKE Furniture Catalog Web application are now all documented.

7. Hovering your cursor over the unit's red X shows only one error. Using the quick fixes, host the JKE Furniture Catalog Web app on the Internet Information Server 6.0.

After using the quick fixes, the diagram will look like Figure 20.
Figure 20. JKE Furniture Catalog Web app hosted on ISS 6.0 and depending on FURNDB

Host the ASP.NET server on a Windows server

You have now documented the catalog application in the topology. The last step is to document it being hosted on the Windows server where it will run.

The steps of documenting the server and its operating system were described in the Host the database on a Linux server section. Therefore, this section will explain the steps only briefly; see the earlier section for the details.

JKE Furniture runs its Web applications on physical servers that are similar to the database servers.

1. Create an x86 Server unit and name it IBM x3450. Set the machine's CPU Count to 2 and its memory to 2GB., and then update the unit's presentation details so that the two attributes are displayed on the diagram.
2. Next, create a Windows Server 2003 unit and set its hostname attribute to "furn-www.jke.com". The Windows Server unit's Administrator user has an error, because the password is not set; however, because the user accounts are not important to this topology, just delete all of the user account and user group units from the Windows server unit.
3. After deleting the users and user groups, drag the furn-www.jke.com unit onto the recently created IBM x3450 server.
4. Then drag the Internet Information Server 6.0 unit onto the now-nested Windows Server 2003 unit.
5. After that, drag the JKE Furniture Catalog Web App unit onto the Internet Information Server 6.0 unit.

At this point, your deployment topology will have zero errors and is complete. The final diagram should look like Figure 21.
Create a Palette item for the ASP.NET server unit

The deployment topology diagram is finished. The steps documenting the capabilities and requirements of the ASP.NET units were a bit time-consuming, because the software does not include templates for the .NET domain. However, Rational Software Architect makes it easy to harvest reusable units and share them as palette entries for others to use.

The easiest way to harvest and promote a unit to the palette is to right-click on the unit and, from the drop-down menu, select the Add to Palette menu item. However, a better way to harvest and share units is to use a template topology. Template topologies are better because the template topology can be stored and shared through version control systems or through IBM® Rational® Asset Manager.

In this section, you will create two template topologies: one for the Internet Information Server 6.0 unit and one for the ASP.NET 2.0 Web Component unit. Each template will result in a palette item that others can reuse.

Create a template

1. To create a template, create a new Topology from the File > New > Topology menu items. Figure 22 shows values in the New Topology window. These are the key values to set:
   • Name: MicrosoftInternetInformationServer6
   • Source: FurnitureCatalogModels/templates
   • Namespace: com.jke.templates
   • Type: Template Design
   • Add topology to palette item: Checked
2. Then click Next.
Figure 22. New Topology window for creating a palette entry

3. Figure 23 shows the final values and icon for the Add Topology to Palette page. The key values to set on this page are:
   - **Label:** Internet Information Server 6
   - **Stack:** Microsoft
   - **Maintain visual structure:** Checked

4. Click **Finish**.
Create a template topology diagram

A template topology diagram acts basically like a normal topology diagram. The key difference is that when someone uses the palette item from the template, a deep copy that preserves the UI formatting of the whole diagram is created and placed into the other diagram.

The ProductionDeployment topology created earlier has the units needed for this template.

1. Go to the ProductionDeployment diagram and copy the Internet Information Server 6.0 unit by selecting it. Then right-click the mouse and, from the drop-down menu, select the Edit > Copy.
2. Back on the Microsoft Internet Information Server 6 diagram, right-click and, from the drop-down, menu select Edit > Paste.

The copy action created both the Internet Information Server 6.0 unit and the JKE Furniture Catalog Web App unit. Given that this is the template just for IIS, the IIS template should not have the JKE Furniture Catalog Web App unit.
3. Select the Web application's unit and delete it from this template topology by using CTRL + DEL.

After the Web application's unit has been deleted, the diagram will have just the single unit of Internet Information Server 6.0, and it will look like Figure 3.

**Note:**
Rational Software Architect V7.5.1 had a minor bug (which is fixed in V7.5.4) with templates that end with a dot number (for example, 6.0). Therefore, to ease the use of the palette item, rename the unit to just **Internet Information Server 6**.

**Figure 24. Internet Information Server 6 template topology diagram**

Creating a template topology for the ASP.NET Web Component is very similar to those steps.

4. Create a new topology that uses these key values:
   - **Name:** MicrosoftASPDotNET2WebComponent
   - **Source:** FurnitureCatalogModels/templates
   - **Namespace:** com.jke.templates
   - **Type:** Template Design
   - **Add topology to palette item:** Checked
   - **Label:** ASP.NET 2.0 Web Component
   - **Stack:** Microsoft
   - **Maintain visual structure:** Checked

5. Again, the ProductionDeployment topology has the unit needed for this topology, so go to the Production Deployment diagram, copy the JKE Furniture Catalog Web Component unit from the diagram, and paste it into the **Microsoft ASP Dot Net2 Web Component** diagram.

The big difference in harvesting this unit is that the unit needs to be renamed and the requirements need to be adjusted.

6. Rename the unit to **ASP.NET 2.0 Web Component**.

7. After renaming the unit, go to the Properties view's **Requirements** tab. The unit still has the dependency on the FURNDB database. Because this template palette is going to create generic ASP.NET 2.0 Web components, delete the FURNDB requirement by using the Properties view's **Delete Requirement** button.

After making these changes, the template topology diagram will look like Figure 25.

**Figure 25. ASP.NET 2.0 Web Component template topology diagram**
The templates are now ready to use, and the palette items are available in the palette's Local Extensions drawer. These templates can be reused immediately and updated later if enhancements need to happen over time. Remember though, that if the template is updated, the units created by the earlier template are not updated.

If you did not complete all of the exercise's steps, you can download this article's Rational Software Architect sample project (see Downloads).

**Summary**

The deployment topology captures how a solution's software and hardware will be deployed and configured. Deployment topology diagrams are a great way for application development teams and infrastructure teams to share information.

A deployment topology can be created in quickly by using the templates that are included in Rational Software Architect. It does take a little more time to create a deployment topology when the technology domain is not provided; however, you can easily harvest and share these domains so that future physical topologies can be created quickly.
## Downloadable resources

<table>
<thead>
<tr>
<th>Description</th>
<th>Name</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>RSA project in this article</td>
<td>FurnitureCatalogModels.zip</td>
<td>12KB</td>
</tr>
</tbody>
</table>
Related topics

- Download a trial version of Rational Software Architect
- Subscribe to an IBM developerWorks newsletter.
- Download a free IBM product evaluation.
- Build your next app with IBM Cloud Lite. No time limit. No credit card required. Start for free

© Copyright IBM Corporation 2009
Trademarks
(www.ibm.com/developerworks/ibm/trademarks/)