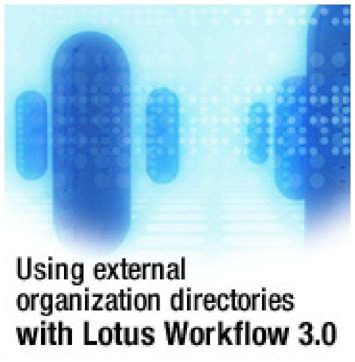
Lotus Developer Domain — The Technical Resource for Lotus software — LDD Today



Level: Advanced
Works with: Workflow 3.0
Updated: 01-Jul-2002

by
Seol Young Park

Lotus Workflow 3.0 gives you increased flexibility in defining your Organization Directory. In previous releases, you were required to store all your person, group, and related information in the Organization Directory database. Workflow 3.0 offers enhanced processing of external resources such as a Domino Directory or an LDAP directory, and you can now define an external custom directory as your Organization Directory. This lets you leverage existing information, maintaining your organization's data with much less effort.

However, you may want to extend this flexibility even further and use another external database format as your Organization Directory. For example, you may have a Notes TeamRoom that contains person and group information that you want to use with Workflow. This article explains how to do this, providing all the code necessary to customize Workflow to define a TeamRoom as the Organization Directory. It also includes a detailed explanation of how the code works so that you can adapt the code to define other database types as your Organization Directory. You can download the complete sample code from the **Sandbox**.

This article assumes you are experienced with LotusScript and are familiar with Lotus Workflow and Domino Designer. To download documentation for these products, see the **Documentation Library**. For the latest product information, see the **LDD Workflow page**.

Organization Directories in Lotus Workflow 3.0

As mentioned, Lotus Workflow 3.0 introduces improved ways to process people information. For example, in previous releases the Workflow Engine might spend a significant amount of processing time evaluating groups to determine whether the names are all members of the organization. But release 3.0 doesn't require Person documents in the Workflow Organization Directory database. This improves performance by skipping many lookups and keeping the database smaller, which in turn, results in fewer maintenance issues. Most importantly, this allows for much better processing of external sources—that is, processing of person information stored outside your Organization Directory.

At the same time, however, Workflow 3.0 does have some limitations:

- An incorrectly entered Organizational Unit (OU) name doesn't result in an immediate error message. Instead, Workflow treats the name as an unknown person within the external directory. Any error message you receive will occur after processing is complete.
- If you add users using a list in a dialog box based on Domino Directories, you can also select names of groups (rather than person names) without receiving an error message. However, these groups will not resolve properly. Again, you will discover this only after processing is complete.
- The Lotus Workflow Engine doesn't accept dynamically specified external group names.

If your organization is typical, your employee information is in constant flux as people come and go, assume different projects and responsibilities, and so on. So when you assign Workflow activities, it's better to use roles, departments, rules, and workgroups rather than individual names. By default, the Workflow Engine looks for this information in the Workflow Organization Directory, where it is available to both the Workflow Engine and Lotus Workflow Architect. If this is the way your site is set up and you only need simple customized access to your external sources, you can use @Relations or @CustomFormula to access that information. For example, you may only need to find the name of an employee's manager or the names of members of a particular department, from a source other than the Workflow Organization Directory. If so, it's possible to do this through the default @Relations feature that comes with Workflow. (For information on how to use @Relations, see the Lotus Workflow tip, **Using Relations for flexible Domino Workflow design**.)

On the other hand, if your organization stores person information in a source other than the default Workflow Organization Directory, a Domino Directory, or an LDAP directory, you'll need to customize Workflow to use the information in this other source. This is what we'll do in this article, using a Lotus TeamRoom as the Organization Directory.

Creating a Resource document in the Organization Directory

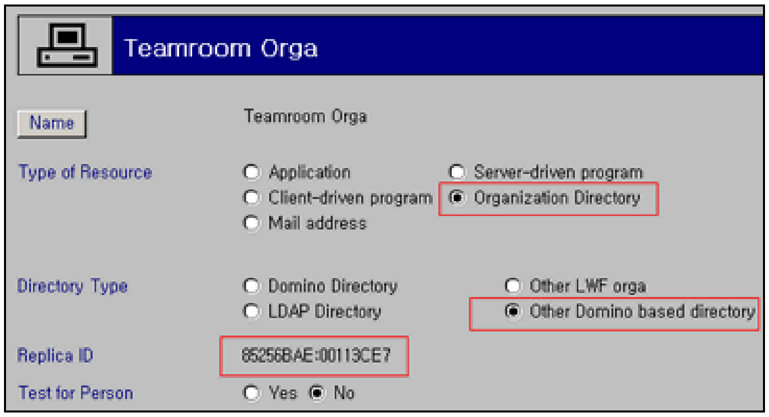
The first thing you need to do is inform the Architect and the Engine where your external organization information database resides. You could define it in the Architect, but then it will not be available for the Engine to evaluate at runtime. Therefore, the best approach is to create *Resource documents* in the Workflow Organization Directory.

As previously mentioned, Workflow 3.0 supports Domino Directories and LDAP directories by default as directory types. To use another type of external source, you must add additional data types to the DirectoryTypeOS field in the Organization Directory's Resource form. In the following example, we add two new types, "Other LWF organization directory | DWFOrgDir" and "Other Domino based directory | OtherDominoDB." Note that when adding these data types, you can enter any name for display, but you must use the appropriate alias from the following list:

- Domino Directory|DD
- LDAP Directory|LDAP
- Other LWF organization directory | DWFOrgDir
- Other Domino based directory|OtherDominoDB

To begin, create a Resource document with Organization Directory as the resource type. Do this by creating a Resource document and selecting "Other Domino based directory" as your directory type:

1. Open the Organization Directory.
2. Choose Infrastructure – Resources in the navigator. Existing resource documents appear in the window on the right.
3. Click the Create Resource Document button. A new resource document appears.
4. Click the Name button and enter a name.
5. Select Organization Directory as the Type of Resource.
6. Select Other Domino based directory as the Directory Type.
7. Enter the Replica ID of your external directory database.
8. Save and close the Resource document.


Teamroom Orga

Name	Teamroom Orga
Type of Resource	Application / Client-driven program / Mail address / Server-driven program / [✓ Organization Directory]
Directory Type	Domino Directory / LDAP Directory / Other LWF orga / [✓ Other Domino based directory]
Replica ID	85256BAE:00113CE7
Test for Person	Yes [✓ No]

Now, both the Architect and the Engine will be able to locate your external directory.

Setting up the Lotus Workflow Architect

After you create the Resource document for the new directory type, you must configure the Lotus Workflow Architect so that it can use the new format. The table below lists the data source formats the Architect can use to obtain information:

Format	**Comments**
Lotus Workflow Organization Directory	Default
Domino Directory	Alternative
LDAP	Alternative
LiSA	**Lotus i-Net Solution** Alternative
DLL	DLL Organization Directory Alternative No mapping file required
Other Notes-based databases	Requires customization including a mapping file
Resource database	Optional

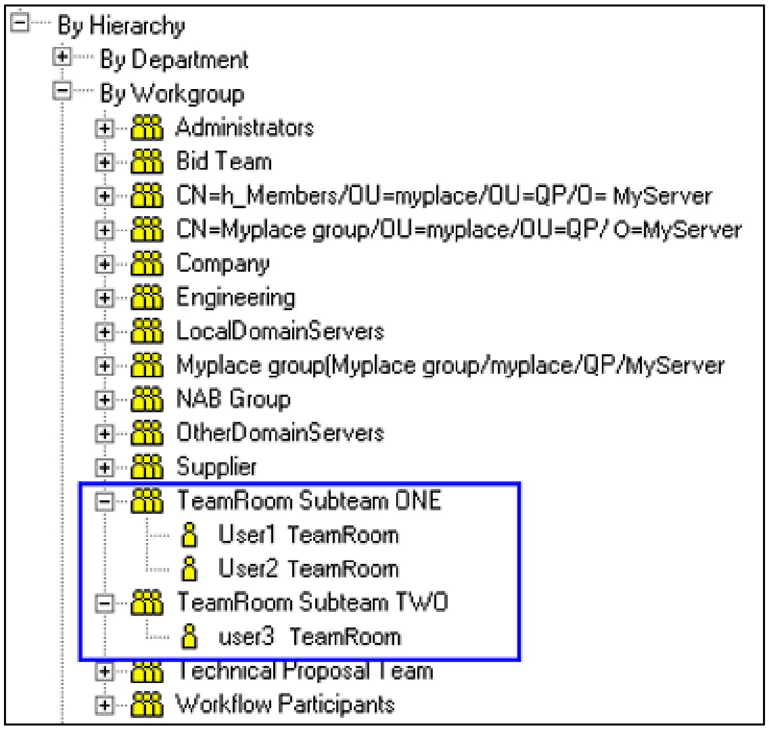
You can determine which Data Source types are available in the Architect by choosing File - Open - Database - Add from the Architect menu. Because you already have specified your external source in the Resource document, you need to tell the Architect how to use this information. You do this by creating a *mapping* file in the Architect directory. This mapping file is an INI file that defines which type of organization units are available and how to find and display them. You can also create separate mapping INI files and place them in the Architect directory or you can append your additional information to an existing mapping file. In our example, we will append to the existing mapping INI file, lwfmap.INI, which ships with Lotus Workflow. (As with any INI file, be sure to back up the original file before editing it.)

Add the following section to the end of the lwfmap.INI file. (You can download this code from the **Sandbox**.)

```
;=== TeamRoomOrga ================================================

[TeamRoomOrga]
Type=TeamRoom
Version=1.0
System=Orga
Platform=Notes
ID=View#SubteamLookup
SupportedTypes=Person,Workgroup

[TeamRoomOrga:Person]
Selection=View#PeopleLookup
SystemName=Field#Who
DisplayName=Formula#@Name([CN];Who)

[TeamRoomOrga:Workgroup]
Selection=View#SubteamLookup
SystemName=Field#STName
DisplayName=Field#STName

[TeamRoomOrga:PersonByWorkgroup]
RelationName=HasMembers#6537
InverseRelation=IsMemberOf
SourceType=Workgroup
TargetType=Person
Selection=Formula#Form ~ "Subteam" & STName ~ "%NAME"
SystemName=FieldList#STMembers
DisplayName=MappingUnit#OrgaObjectByID

[TeamRoomOrga:OrgaObjectByID]
Selection=Formula#@Name([Abbreviate];Who) ~ "%ID"
SystemName=Field#Who
DisplayName=Formula#@Name([CN];Who)
    TypeTag=Formula#@If(Form ~ "Subteam"; "Workgroup"; Form ~ "ParticipantProfile"; "Person";Form)
```

Notice that in this code:

- The TeamRoomOrga section specifies that this mapping file is intended for the Organization Directory, and the external database is identified through the unique design elements defined by the ID. The Architect uses the ID to locate the data source. This code supports two types of organization units, Person and Workgroup. Each organization unit section defines a view used to find the units and field names to be used for display or by the system. This will display your business objects under the general People/Group objects, and allow you to see them by hierarchy, as shown in the following screen.
- The TeamRoomOrga: PersonByWorkgroup and TeamRoomOrga: OrgaObjectByID sections display your organization units by hierarchy. In our example, TeamRoomOrga: PersonByWorkgroup displays people selected (through a formula) by their Workgroup.
- Relations (dependencies between business objects) are defined by the variables RelationName, InverseRelation, SourceType, and TargetType. The value 6537 in the RelationName variable instructs the internal resource ID in Workflow to get the correct string to display for the relation. If you add your own new entry, you should omit the number. You can check the relations between business objects in the Object Browser by right-clicking and choosing Show Relation from the menu.
- The MappingUnit defined by DisplayName isn't necessary if you organization type has single organization unit.
- %ID in the TeamRoomOrga:OrgaObjectByID will be replaced with the selected objects UniqueID.
- The variable TypeTag determines the type of the organization unit when you have a mapping rule that applies to multiple object types.
- At runtime, the tilde character (~) in formulas is replaced with an equal sign (=) and the %NAME variable is replaced with the selected objects DisplayName.


```
By Hierarchy
    By Department
    By Workgroup
        Administrators
        Bid Team
        CN=h_Members/OU=myplace/OU=QP/O= MyServer
        CN=Myplace group/OU=myplace/OU=QP/ O=MyServer
        Company
        Engineering
        LocalDomainServers
        Myplace group|Myplace group/myplace/QP/MyServer
        NAB Group
        OtherDomainServers
        Supplier
        TeamRoom Subteam ONE
            User1 TeamRoom
            User2 TeamRoom
        TeamRoom Subteam TWO
            user3 TeamRoom
        Technical Proposal Team
        Workflow Participants
```

You can now create your sample process design using the new people and groups and examine your Process Definition database. You should see that Workflow now generates with a new format. For more information on how to customize your own mapping files, refer to **Lotus Workflow Software Developer's Kit** or review the default lwfmap.INI file.

Customizing the Workflow Engine for your external directory

Now that you've defined the new directory type in the Resource document and configured the Architect accordingly, it's time to customize the Workflow Engine itself.

Understanding the key format and assigning group names dynamically

By default, Workflow 3.0 supports Domino Directory as an external directory type. You can use a Domino Directory as an external data source by creating a Resource document for it. If you're using a Domino Directory as an external Organization Directory, and you assign a group as an activity owner, the Architect will activate information for the Engine with the following key format in the Process Definition database:

NABGroup/WFN=\"NABGroup\"/WFT=Workgroup/WFD=\"<resource_name_for_your_Domino_Directory>\"

The key format consists of five items with delimiters:

ID=<ID of the key>
/WFN=<DisplayName>
/WFT=<Organization Unit Type, workgroup in the example>
/WFD=<Directory Name , resource name for your external source>
/WFU=<YES, to define whether this organization unit is a user>

The ID is required; the other four items are optional and generated only if necessary. In the example above, the Engine interprets the organization unit as a group in a Domino Directory, with NABGroup as the ID and display name.

What happens if you try to use external group names dynamically? For example, you may want your activity owner to be decided at runtime rather than at process design time. Assume you have a formula such as @If (FieldA= "" ; "NABGroup1" ; "NABGroup2") for your activity owner in the Architect. When the Architect populates it, the Engine will get only the formula itself to evaluate at runtime, resulting in "NABGroup1" or "NABGroup2," without any resource information. The Engine will therefore, try to evaluate membership against the default Workflow Organization Directory and not find the names of the people you need.

Suppose that you have a keyword field with aliases, for instance:

"NABGroup |
NABGroup/WFN=\"NABGroup\"/WFT=Workgroup/WFD=\"<resource_name_for_your_Domino_Directory>\"

Or simply:

NABGroup | NABGroup/WFD=\"<resource_name_for_your_Domino_Directory>\"

When a user selects the NABGroup in the keyword field, the field alias with the key format is saved and will provide the detail information to resolve the group. This way you can use external group names dynamically. Also, you can do this by using the event QueryExpandParticipants. However, this may require more effort to customize.

Customizing the Workflow Engine
You have set up the Architect to understand your external source, and you've activated your processes with the key format in your Process Definition database. Now, it's time for you to make the Engine accept external directories other than the Domino Directory for the runtime evaluation.

Continuing with the TeamRoom example, you next need to have the Engine load your external directory using the QueryGetOrganizationDirectory event (instead of the default directory) and then have your custom code resolve your participants list. When the Engine evaluates participants, it reads the string or the formula passed by the Architect and creates a Workflow key object (WFOrgUnitKey). This key object retrieves properties such as ID, UnitType, DisplayName, and Directory. It then checks whether or not the directory object has been loaded. If not, it looks for Resource documents you've created to load the directory, and then initiates the event QueryGetOrganizationDirectory. This is where we can make use of our custom directory.

In the example, you'll notice several Workflow classes used as your base classes. This code sample is powerful and flexible. In most cases, however, you will need only a subset for your implementation. Let's take a look step-by-step.

Step 1: Create a Lookup view in the TeamRoom database
First, you create a Lookup view in the TeamRoom database. This view allows lookup for all units. If you already have a view in your custom database that does this, you can skip this step.

1. In Domino Designer, make a copy of the (SubteamLookup) view that comes with the TeamRoom database.
2. Rename the new copy AllLookup.
3. Add the following view selection formula:

   SELECT (form = "DefSubTeam" | form = "Subteam" |form = "ParticipantProfile" | form = "Participant Profile") & Status = "1"

4. Add the following first column formula:

```
@If (Form = "DefSubTeam":"Subteam"; STName ;form = "ParticipantProfile":"Participant Profile";Who ;"" )
```

Note that you can also copy the AllLookup view from the ExternalOrgaSample.nsf database provided in the **Sandbox**. To do this, open ExternalOrgaSample.nsf in Domino Designer, copy the AllLookup view, and paste it into your TeamRoom database.

Step 2: Create two customized classes and add them to the OS Application Events script library
You must now create two custom classes, SimpleOrgUnit and TeamRoomDirectory, and add them to the OS Application Events script library in your TeamRoom. The customized class SimpleOrgUnit is derived from WFOrgUnit to work with the TeamRoom example. WFOrgUnit supports more complex scenarios including subgroups and substitutes. We've included some of those properties for demonstration, although they will not be used in this TeamRoom example. The second custom class is TeamroomDirectory. This is derived from the DWFDirectory class. The New method will load the custom directory using the directory information from the key and the ViewName property. In this example, we've overridden the GetOrgUnit methods and substituted our function.

1. Open your TeamRoom in Domino Designer, and click Script Libraries.
2. Click OS Applications Events.
3. Click (Declarations), and add the following code

```
Class SimpleOrgUnit As WFOrgUnit
    Public bIsInitialized As Integer
    Public bIsValid As Integer
    Public vDirectMembers As Variant  ' this contains the actual names for the direct members in the group
    document.
    Public vSurrogateMembers As Variant
    Public vSurrogateSubgroups As Variant
    Public bAllowSubSequentSurrogation As Integer
    Public bIncludeSubgroupsByDefault As Integer
    Public vSubgroups As Variant
    Public bMixedMembers As Integer

    Sub new (Key As WFOrgUnitKey),WFOrgUnit(key)
    End Sub

    Property Get IsInitialized As Integer
        IsInitialized=Me.bIsInitialized
    End Property

    Property Get IsValid As Integer
        IsValid=Me.bIsValid
    End Property

    Property Get SurrogateMembers As Variant ' list of string (UserKeys)
        SurrogateMembers=Me.vSurrogateMembers
    End Property

    Property Get SurrogateSubgroups As Variant ' list of strings (OrgUnitKeys)
        SurrogateSubgroups=Me.vSurrogateSubgroups
    End Property

    Property Get AllowSubSequentSurrogation  As Integer
        AllowSubSequentSurrogation=Me.bAllowSubsequentSurrogation
    End Property

    Property Get IncludeSubgroupsByDefault As Integer
        IncludeSubgroupsByDefault=Me.bIncludeSubgroupsByDefault
    End Property

    Property Get DirectMembers As Variant 'name list of direct members
        DirectMembers=Me.vDirectMembers
    End Property

    Property Get Subgroups As Variant
```

```
            Subgroups=Me.vSubgroups
        End Property

        Property Get MixedMembers As Integer
            MixedMembers=Me.bMixedMembers
        End Property

    End Class

    Class TeamroomDirectory As DWFDirectory

        Private Property Get ViewName As String
            'This is a view to be used for look up for people and groups in the TeamRoom database.
            'You can find the view used in the sample NSF.
            ViewName = "AllLookup"
        End Property

        Sub new(DirName As String, Source As Variant) , DWFDirectory(DirName, source)

        End Sub

        Function GetOrgUnit(key As WFOrgUnitKey) As WFOrgUnit
            If Not Me.IsInitialized Then Exit Function
            Set GetOrgUnit = TeamroomDirectory_GetOrgUnit(Me, key)
        End Function

    End Class
```

This code is also available in the ExternalOrgaSample.nsf database provided in the **Sandbox**. To copy it to your TeamRoom:

1. Open ExternalOrgaSample.nsf in Domino Designer.
2. Click Script Libraries.
3. Click OS Application Events.
4. Click (Declarations).
5. The entire (Declarations) code appears in the programming pane. You can copy this code and paste it into the (Declarations) section of the OS Application Events script library in your TeamRoom.

Step 3: Add the customized GetOrgUnit function to OS Application Events script library

The GetOrgUnit function will find the organization units from the custom directory and return the actual names. This example includes extra code (which we've identified through a comment) to convert TeamRoom names to canonical names. If your external database uses canonical names, you can omit this extra code.

1. Open your TeamRoom in Domino Designer, and click Script Libraries.
2. Click OS Applications Events.
3. Click (Declarations), and add the following code:

```
    Function TeamroomDirectory_GetOrgUnit(pMe As TeamroomDirectory, okey As WFOrgUnitKey) As
    SimpleOrgUnit

        On Error Goto errorhandler

        Dim ou As SimpleOrgUnit
        Dim doc As Notesdocument
        Dim oName As NotesName
        Dim i As Integer
        Dim vTemp As Variant

        Set TeamroomDirectory_GetOrgUnit = Nothing
        Set doc = pMe.View.GetDocumentbyKey(okey.ID, True)    'look for the subteam(group) documents in the
        (AllLookup) view.
        If (doc Is Nothing) Then
            ' document cannot be found. return "nothing" and create no error. another dir may be searched for
            this unit
            Exit Function
        End If

        If doc.HasItem("STMembers") Then
            'This is a subteam(group) document and the field contains user names.
              'Create a new SimpleOrgUnit object and transfer all properties and unload document
            Set ou= New SimpleOrgUnit(okey)
            Redim vTemp(Ubound(doc.STMembers)) As String
            For i=0 To Ubound(doc.STMembers)
                Set oName = New NotesName(doc.STMembers(i))
                If oName.IsHierarchical Then
                    vTemp(i) = oName.Canonical
                End If
            Next
            ou.vDirectMembers= Fulltrim(vTemp) 'Assign the actual names to DirectMembers since we know
            there is no subgroups inside.
            ou.bIsInitialized  = True
            ou.bIsValid = True
        End If

        Delete doc

        Set TeamroomDirectory_GetOrgUnit = ou   ' Return SimpleOrgUnit

        Exit Function
    errorhandler:
        Print "Internal Error (" + Error$ + ")"
        Exit Function
    End Function
```

This code is also available for you to copy in the OS Application Events script library in ExternalOrgaSample.nsf.

Step 4: Add code to QueryGetOrganizationDirectory event to load your custom directory
The following code locates your custom directory, based on the resource name you specified in the Resource documents. This is TeamRoom Orga in our example. Although our example offers one way to do this, you may prefer to define a field to serve this purpose in your Resource document. This has the advantage of including the resource name in the document, where you can edit and change it later.

1. Open your TeamRoom in Domino Designer, and click Script Libraries.
2. Click OS Applications Events.
3. Click (Declarations).
4. Click QueryGetOrganizationDirectory, and add the following code:

```
    Private Function QueryGetOrganizationDirectory_(Continue As Integer , DirName As String, DirectoryType As
    String, ResourceDocument As Notesdocument, OrgDir As Variant )
        If DirName Like "*Teamroom*" Then 'It's been specified in your resource document
            Set OrgDir = New TeamRoomDirectory(DirName, ResourceDocument)
            continue=False
        End If
    End Function
```

As with the previous examples, this code is available in the OS Application Events script library in ExternalOrgaSample.nsf.

After you complete all the steps included in this section, you should be able to assign your TeamRoom people and groups in the Workflow Architect, as well as Workflow Engine.

Extending the power of Lotus Workflow

As we've shown above, Lotus Workflow is both powerful out-of-the-box and highly flexible and customizable. We've shown you how to extend Lotus Workflow to accept external directories as the Organization Directory. This includes setting up the Architect to understand your external source and activating your processes with the key format in your Process Definition database. We also explained how to instruct the Engine to accept an external directory other than the Domino Directory for the runtime evaluation. Our example uses a TeamRoom as an external directory of organization information, but with a little creative adapting, you can apply our example to other directory types as well. Try your ideas and share your experiences with us!

ABOUT THE AUTHOR

Seol Young Park joined Lotus/IBM in 1994. She's been a member of the Lotus Workflow team since 1999, serving as a developer for the Lotus Workflow Engine. Previously, Seol Young worked on multiple projects for the International Product Development team.