



Level: Advanced
Works with: Domino Off-Line Services
Updated: 04-Nov-2002

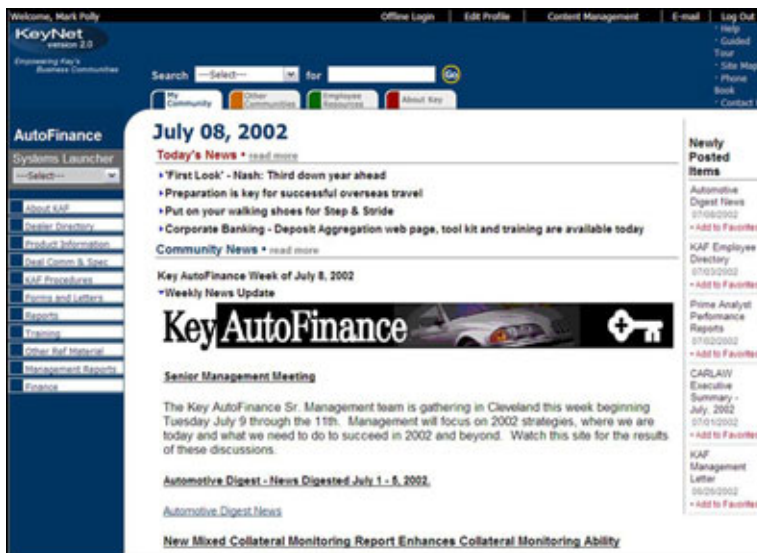
Selectively replicating DOLS subscriptions: One customer's story

by
 Mark Polly
 and Mark Bratt

This is the story of a Domino Off-Line Services (DOLS) customer who had a problem replicating data to its users. As administrators and application developers with DOLS experience have found, replicating only the data that users need can be tricky because while selective replication works well in the Notes client, DOLS subscriptions are accessed not through the Notes client, but the Web browser. So how did one customer find a solution to this problem?

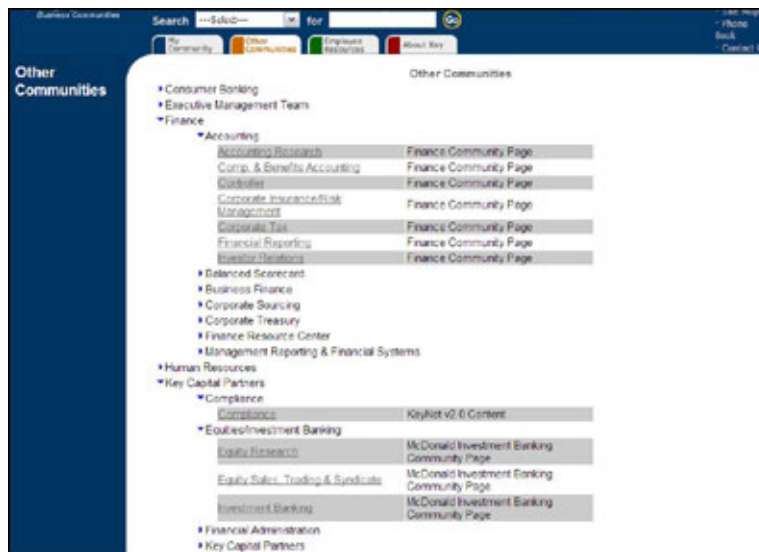
The KeyBank Story

KeyNet v2.0 is the newest company internal Web site for KeyBank. This Domino-powered Web site is designed to group users into communities along lines of business. Each community has its own homepage maintained by Content Managers in that community. When a user logs into KeyNet v2.0, the community's homepage is displayed, and from the homepage, the user can link to the community's documents, view email, and navigate to other communities.



Currently, KeyBank has over 40 communities using KeyNet v2.0. We store all community content in one database to allow users to cross-link to documents created by other communities. The KeyNet v2.0 site consists of five interconnected databases. In addition to the content database, four other databases are used to provide hit

tracking, administration, images, and user profiles for the site. The following screenshot shows the list of communities accessible from the site.



Most employees access KeyNet v2.0 via the company's intranet. However, some of the sales people require mobile access to the application. The application development team decided to use Domino Off-Line Services (DOLS) to enable mobile users to access the application off-line. The size of the site is over 5 GB, too large for users' laptops. So we needed a way to enable users to download only their community's documents, thus reducing the local, off-line application to 50-200 MB.

Of course, Notes is known for selectively replicating data to your local PC for use off-line. It is relatively straightforward to use the Notes client to enter a replication formula for each user. But with DOLS, there is no Notes client. Access to the replication formula is through the server, not through the individual PCs. How could we write a replication formula that would execute from the server and identify each user uniquely (that is, get their community name)?

Now you might think, aha, let's use the @DBLookup formula to look up the user's name in a profile database and grab a field value to limit the replication. In fact, you can type in a nice @DBLookup formula in the Replication Settings dialog box, but the formula will not return results because @DBLookup is not a valid formula for replication settings. Neither is @DBCColumn valid for the same settings.

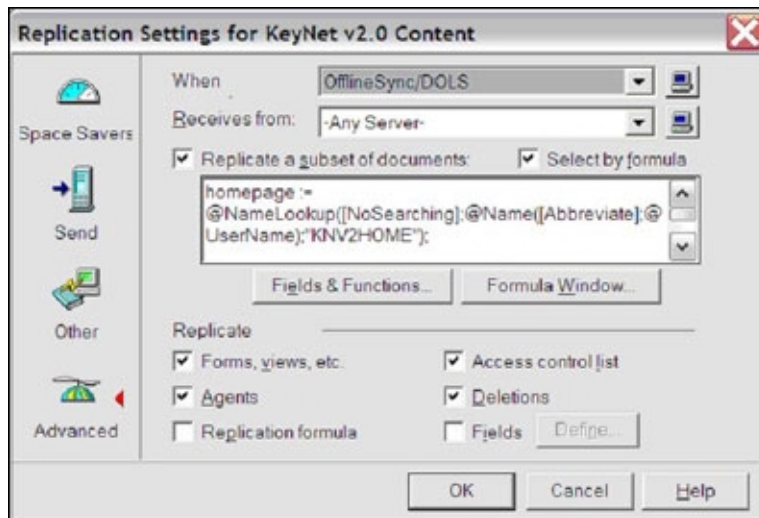
We researched other possible solutions. For instance, we considered splitting the information into different databases, based on the different communities. This idea would allow us to point a user to a specific database to replicate, without needing a replication formula. However, we rejected the idea for two reasons:

1. The system would become too complex.
With a site that already consists of five interrelated databases, splitting the site to support 40 separate communities would require 200 databases (40 communities x 5 databases) to allow selective replication by community.
2. Communities share information.
Because a good deal of content is available to multiple communities, we would have to maintain separate replication formulas for each database. Imagine trying to code a hyperlink formula dealing with this many back-end databases!

The solution we came up with was to use a new formula in Domino 5 called @NameLookup. In this article, we describe how we implemented our solution by modifying the Person document in the Domino Directory with an agent and using the @NameLookup formula to selectively replicate documents to our DOLS users. This article assumes familiarity with Domino Off-Line Services and knowledge of the Notes formula language and LotusScript.

Replication formula for DOLS users

When setting up the replication formula for DOLS, we selected the DOLS subscription, then used the Replication Settings dialog box in the Notes client.



In the preceding dialog box, notice that OfflineSync/DOLS is selected in the When field, and Any Server is selected in the Receives from field. (Both should be selected by default.)

This dialog lets us specify a subset of documents to replicate. Normally, we might limit the off-line user to replicate only documents he/she created using @Author. But what if we want to replicate Marketing documents for all marketing people, and IT documents for all IT people, regardless of author? The replication settings allow us to specify Any Server, or enter a specific server name to apply the settings to *all* DOLS users equally. Without hard-coding each user name into the Receives from field, there is no obvious way to selectively replicate documents that do not directly reference a user's name.

DOLS 5.0.9 and later lets us set selective replication for specific users using the Replication Setting dialog box. On the Advanced tab of the dialog box, we can specify a user in the When field to customize replication for that user, but with over 1,000 replicating users, this option becomes unmanageable. Using our solution, we can provide a unique replication formula for each user with only one setting on the server. For more information about this selective replication option in DOLS, see the [Domino Administrator's Help](#).

Luckily for us, the @NameLookup formula works in a selective replication formula. Introduced in Domino 5, the @NameLookup formula looks up information from the Domino Directory and returns a field from the Person document. If we write the user's community name value to the Domino Directory, we can use @NameLookup to retrieve the community name value using the following replication formula:

```
homepage := @NameLookup([NoSearching];@Name([Abbreviate];@UserName);"KNV2HOME");
```

```
SELECT (form="DocInput" & @IsMember(homepage; AdminLOB: DBAdd)) | @IsMember(AdminLOB; "About  
Key": "Employee Resources": "KeyNet v2.0 Content": "Help": "Guided Tour") | form = "DOLSOOfflineConfiguration" |  
form = "LkupDoc" | form = "Fintranet" | form = "Doclink"
```

The first line performs an @NameLookup of the current user. The first parameter—[NoSearching]—tells @NameLookup to search only the first directory for the user's name. The @NameLookup recognizes the hierarchical name in the abbreviated format, so we use the @Name formula to abbreviate the user's name. Even though this formula is configured on the server, when the replication formula executes, it runs locally on the user's machine to return the current user's ID. The parameter KNV2HOME is the name of the field in the Domino Directory containing the value we want returned.

The following screenshot shows how we used the @NameLookup function to show the home community—AutoFinance—in the following example.



In our case, we added the KNV2HOME field to the Person document through an agent, which we describe later in this article. The value of KNV2HOME is actually the name of the user's home community. When the @NameLookup formula executes, our homepage variable is set to the name of the user's home community.

The second line is a more familiar SELECT statement. This statement selects documents created with the DocInput form, which resides in the content database, and whose fields AdminLOB or DBAdd contain the value of the variable homepage. In other words, the SELECT statement limits the documents selected to those associated with the user's home community. The rest of the formula gets other types of documents, but that is unimportant for this discussion. When the SELECT statement executes, the replication engine replicates only documents that belong to our user's community and not the thousands of other documents.

Defining the home page with user profiles

How does the value of the user's home community end up in the Domino Directory? Each user must complete a user profile in the profile database, one of the five databases that constitute the KeyNet v2.0 site. Instead of using database profile documents, we created a custom user profile database. The main reasons for creating this database were:

- Database profiles are more difficult to manage than documents, which can be easily exposed in a view.
- With over 15,000 anticipated users, we wanted to keep these documents separate from our content database.

The profile tells the system to which community the user belongs. When the user accesses the KeyNet v2.0 site, the user profile determines the correct community home page to display. This provides a personalization of the site for each user.

Users authenticate with the site using their Notes IDs and passwords. We use the Domino Single Sign-On (SSO) feature to automatically log users onto the site if they access the site from another internal Domino application. Because of our corporate policy to keep changes to the Domino Directory minimal, we didn't modify the design of the Domino Directory to act as our user profile database. The following screenshot shows a user profile document.

Updating the Domino Directory with the WebQuerySave agent

Because the user profile stores the information that associates a user with a community, we use an agent to update the Domino Directory with the community name whenever the user changes his or her profile. After the agent updates the Domino Directory with the community name, the selective replication formula used by DOLS automatically grabs the appropriate community information.

The user profile form calls this agent in its WebQuerySave event. The agent code, written in LotusScript, is as follows. (To view this code without text comments, see the sidebar "[LotusScript code for WebQuerySave event agent](#).")

In the first section of code, the agent is doing basic initialization—creating the objects:

```
Sub Initialize
On Error Goto ErrorHandler
Dim s As New NotesSession
Dim db As NotesDatabase, backenddb As NotesDatabase, nab As NotesDatabase
Dim note As NotesDocument, userdoc As NotesDocument
Dim backendview As NotesView, nabview As NotesView
Dim username As NotesName
Set note = s.DocumentContext
Set db = s.currentDatabase
```

The next four lines retrieve field values from the user profile document:

```
lob$ = note.lob(0)
group$ = note.group(0)
div$ = note.division(0)
backendname$ = note.BackEndDBName(0)
```

The variable backendname\$ stores the value of our administration database, another of the five databases that comprise the site. This database maps a community's home page to the hierarchy (lob, group, division). Instead of storing the home page name in a user's profile, we look it up dynamically. This allows us to quickly route users to a new home page if there is an organizational change.

We then create the object for the administration database:

```
'lookup the community home page in the backend db
Set backenddb = s.GetDatabase( db.server, backendname$, False)
```

The following line retrieves the definition document corresponding to the user's hierarchy:


```
Set commdoc = backendview.getDocumentByKey(lob$ & "-" & group$ & "-" & div$, True)
```

And this line retrieves the value of the user's home page:

```
homepage$ = commdoc.homepage(0)
```

After we retrieve a home page value, it is time to find the user's Person document in the Domino Directory. These lines open the Domino Directory and find the user's Person document:

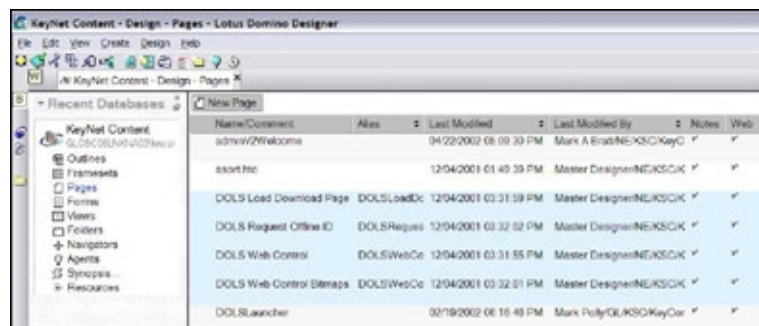
```
Set nab = s.getdatabase("", "names.nsf", False)
Set nabview = nab.getView("$VIMPeople")
Set username = New NotesName(note.ProfileName(0))
Set userdoc = nabview.GetDocumentByKey( username.Abbreviated, True)
```

If the agent finds the document, the home page value is written to a field called KNV2Home and the Person document is saved. The KNV2Home field is used by the replication formula to select the user's community. Unlike adding the user profile form to our Domino Directory, we can write the field KNV2Home to the Domino Directory because it does not affect the design of the directory (which as mentioned earlier, is not allowed by our corporate policy). In essence, we have created a hidden field in the Person document for every KeyNet v2.0 site user. It's a nice feature that lets us adhere to corporate policy, but still implement the functionality that we need in the Domino Directory.

Implementing DOLS in the application

The application development team used the Bruce Hitchcock article "[Domino Off-Line Services: An Administrator and Developer's Guide](#)" to DOLS-enable the KeyNet v2.0 application. The article explains how to copy DOLS design elements into your application to get DOLS to work with your application.

The following screenshot shows the DOLS pages in our content database.



The screenshot shows the Lotus Domino Designer interface with the 'KeyNet Content' database open. The 'Recent Databases' list on the left includes 'KeyNet Content' and 'DOLSUNV202KeyC'. The main window displays a table of design elements:

Name/Comment	Alias	Last Modified	Last Modified By	Notes	Web
adminV2Welcome		04/22/2002 08:00:30 PM	Mark A BrabNE-KSCKeyC	✓	✓
sortfile		12/04/2001 01:40:39 PM	Master DesignerNE-KSCKeyC	✓	✓
DOLS Load Download Page	DOLSLoadDC	12/04/2001 03:31:59 PM	Master DesignerNE-KSCKeyC	✓	✓
DOLS Request Offline ID	DOLSRquest	12/04/2001 03:32:02 PM	Master DesignerNE-KSCKeyC	✓	✓
DOLS Web Control	DOLSWebCo	12/04/2001 03:31:55 PM	Master DesignerNE-KSCKeyC	✓	✓
DOLS Web Control Bitmaps	DOLSWebCo	12/04/2001 03:32:01 PM	Master DesignerNE-KSCKeyC	✓	✓
DOLSLauncher		02/19/2002 06:10:48 PM	Mark PollyOLKSCKeyC	✓	✓

One of the design elements we modified was the DOLSLauncher page. We customized the page by replacing the default HTML with user directions on how to synchronize with the application. The community name is displayed with the <compute text> formula:

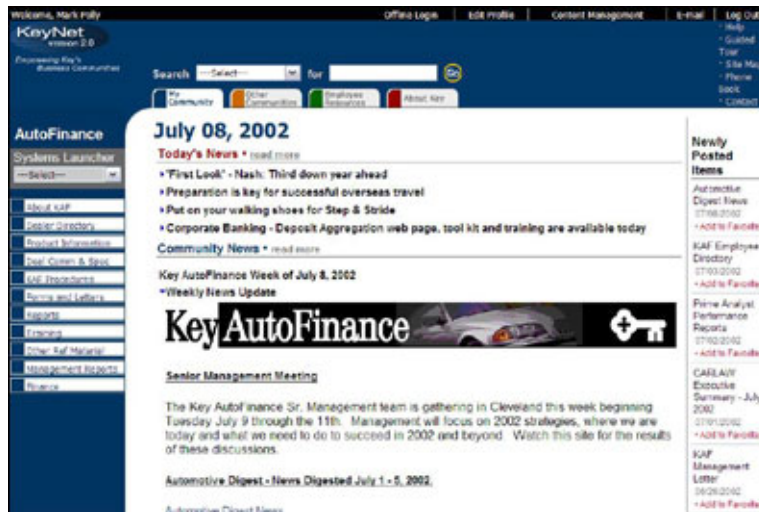
```
@NameLookup([NoSearching]; @Name([Abbreviate]; @UserName); "KNV2HOME")
```

As stated earlier in the article, we implemented Domino Single Sign-on in KeyBank's Notes/Domino environment. If a user initiates replication from a Web browser, she/he has to sign on twice. But if the user initiates replication from the DOLS application in the Notes client, the password is stored locally, so there's no need to sign on again to replicate. However, Single Sign-on caused a problem because we cluster our servers. DOLS with Domino 5 (which we're using) doesn't support Single Sign-on with clustered servers. We solved the problem by partitioning one server into two logical servers. We didn't enable Domino Single Sign-on on the DOLS-enabled server, but the other clustered server is Single Sign-on enabled. So when users first replicate the site, they have to sign on again to download the iNotes Sync Manager and their community's content. Thereafter, if a user initiates the synchronization from the desktop, the user won't have to sign in again because DOLS stores the password locally.

Summary

The result of all this work is to provide an off-line environment that represents most of the information users might

need (all documents in their community), while limiting the amount of disk space required. The following screenshot shows the KeyNet v2.0 off-line screen, which seems an appropriate way to end this customer's story.



ABOUT MARK POLLY

Mark Polly is a Technical Architect with Meritage Technologies, Inc. Meritage is an employee-owned technology consulting company that provides professional services to Global 3500 and Public Sector organizations. Mark is a certified Lotus Notes developer and has worked with Lotus Notes since Release 1. Mark has spent the last six years consulting with a variety of companies on Notes, Domino, and other technology projects.

ABOUT MARK A. BRATT

Mark A. Bratt is a Lead Application Systems Programmer with Key Technology Services. Key Technology Services is a division of KeyCorp, Inc. that provides technical solutions and support to the different lines of business within KeyCorp. Mark started programming in Lotus Notes Release 4 and has worked on various types of Notes projects. Mark has worked at Key Technology Services for the last six years.



LotusScript code for WebQuerySave event agent

```
Sub Initialize
On Error Goto ErrorHandler
Dim s As New NotesSession
Dim db As NotesDatabase, backenddb As NotesDatabase, nab As NotesDatabase
Dim note As NotesDocument, userdoc As NotesDocument
Dim backendview As NotesView, nabview As NotesView
Dim username As NotesName
Set note = s.DocumentContext
Set db = s.currentDatabase
lob$ = note.lob(0)
group$ = note.group(0)
div$ = note.division(0)
backendname$ = note.BackEndDBName(0)
'lookup the community home page in the backend db
Set backenddb = s.GetDatabase( db.server, backendname$, False)
If backenddb.isOpen Then
Set backendview = backenddb.getView("LOBHomePages")
If Not(backendview Is Nothing) Then
Set commdoc = backendview.getDocumentByKey(lob$ & "-" & group$ & "-" & div$, True)
If Not(commdoc Is Nothing) Then
homepage$ = commdoc.homepage(0)
'now write the homepage value to the nab for this user
Set nab = s.getdatabase("", "names.nsf", False)
Set nabview = nab.getView("($VIMPeople)")
Set username = New NotesName(note.ProfileName(0))
Set userdoc = nabview.GetDocumentByKey( username.Abbreviated, True)
If Not(userdoc Is Nothing) Then
userdoc.KNV2Home = homepage$
'save the person doc
Call userdoc.save(True, False)
End If
End If
End If
End If
'let the profile doc save by doing nothing
Exit Sub

ErrorHandler:
MessageBox("Error in WriteCommunityToNab @ " & Erl & " error = " & Error)
Exit Sub
End Sub
```