



Notes to SharePoint Migration without Breaking the Bank

Dealing with Complex Applications

Prepared by
Redmond Media Group

Contributing Author
Curtis Kelly, Senior Product Manager,
Quest Software

Sponsored by

 **QUEST
SOFTWARE**
www.quest.com

Table of Contents

Abstract	1
Introduction	1
How a Lotus Notes Environment Differs from a Microsoft Environment.....	1
How a Notes Application Environment Differs from SharePoint	1
Migrating the ‘Easy Eighty’	1
Overview	1
Step 1: Inventory Your Notes Applications and Assign a Sponsor for Each	2
Step 2: Categorize Your Notes Applications	2
Step 3: Analyze Your Notes Applications	2
Step 4: Choose Appropriate Tools	3
Step 5: Perform a Proof of Concept.....	3
Establish Partnerships	3
Create a Migration Test Bed	3
Select the Applications for the POC	3
What You Should Learn from the POC	4
Step 6: Perform the Migrations	4
Rebuilding the ‘Tough Twenty’	4
Introduction	4
Using Native Tools.....	4
Rapidly Rebuild Notes Applications with Third-Party Tools	4
Reduce Development Time and Cost	5
Deliver a “Notes-like” Experience for your Application Users	5
Deliver Functionality not Available with Native SharePoint.....	6
Conclusion	7
About the Author	7

ABSTRACT

A migration from Lotus Notes and Domino to a Microsoft environment involves more than just moving your email and calendar data to Microsoft Exchange Server; almost all other data and applications must also be migrated to a SharePoint environment. That's where the fun starts. Most standard Notes applications, such as document libraries and Team Rooms, will map easily to existing SharePoint templates, and even some custom applications can be migrated with limited configuration using out-of-the-box SharePoint features and third-party migration tools. However, advanced applications that have taken years to develop can take significant effort to redevelop in a SharePoint environment—and these advanced applications are usually the ones you rely on every day and therefore the ones you need to migrate the most. While these applications comprise 20% or less of your Notes inventory, they will definitely require 80% or more of your migration effort.

Don't worry. You can rebuild complex Notes applications without the need for costly custom development. All you need is the right approach and the right set of tools. Learn more in this white paper, and make your migration a complete success.

INTRODUCTION

How a Lotus Notes Environment Differs from a Microsoft Environment

A successful migration from Lotus Notes to Microsoft requires understanding some fundamental differences between the two environments. Notes is an all-in-one environment that supports not only email and calendaring functions but also collaboration, workflows and document management. To achieve the same functionality in the Microsoft world, whether in-house or in the cloud, you need to bring together a set of different tools (see **Figure 1**). In house, you will need to deploy and configure each of these technologies; if you use the cloud, you can simply rely on Microsoft Office 365 to provide the entire suite of tools.

In particular, in a Notes-to-Microsoft migration, you move all your Notes email and calendar data into Microsoft Exchange, and you move all of your Notes applications into SharePoint.

How a Notes Application Environment Differs from SharePoint

While email migration is relatively straightforward, a successful migration of Notes applications requires an understanding of the differences between a Notes application environment and SharePoint. While Notes can use either the Web or a rich client to provide application functionality, SharePoint is a Web-only

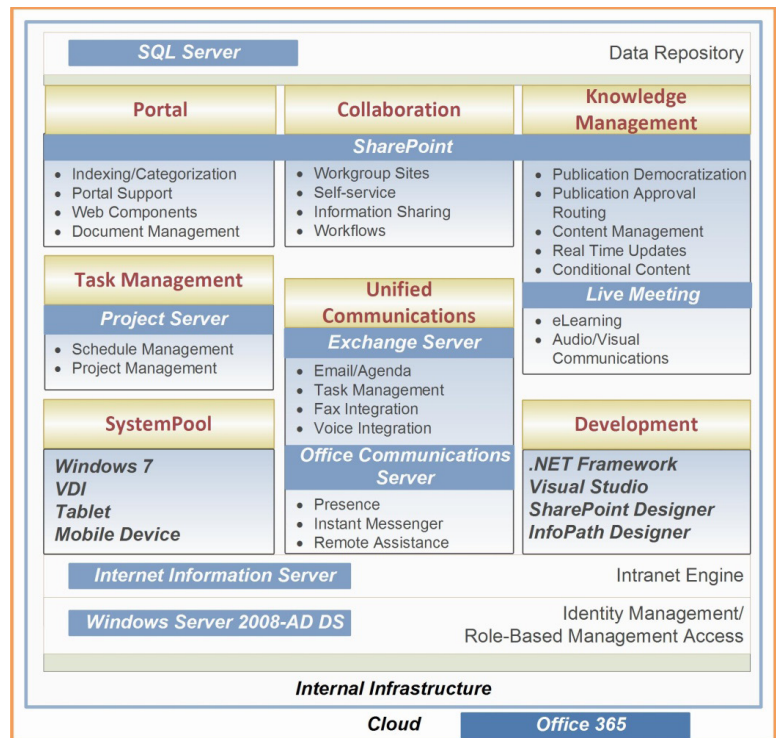


Figure 1. A Microsoft-based infrastructure includes multiple components.

environment. Therefore, your converted applications will gain the flexibility of the Web but they will also have a completely different interface—and this may affect functionality. You'll also have to learn a whole new way of programming and a whole new set of programming tools; for many Notes developers, this is the biggest hurdle in the move from Notes to Microsoft.

MIGRATING THE 'EASY EIGHTY'

Overview

In the Notes world, most development was focused on building fancy forms and views, and most developers got by with canned actions and Notes formulas for simple data validation, scripting of button events, and background processing of documents. The number of people who could write complex LotusScript or Java agents was relatively small, and organizations resorted to that level of coding only when needed.

Therefore, migrating most of your Notes and Domino applications—probably around 80 percent of them—will be fairly straightforward. The process should include the following steps:

- 1. Inventory your Notes applications and assign a sponsor for each** – You should prepare a complete inventory of all of the applications in your Notes environment and identify a sponsor for each application.
- 2. Categorize your Notes applications** – Not all apps are the

same or involve the same level of complexity during migration, so categorization is an essential step of the process.

- 3. Analyze your Notes applications** – You need to understand how each application works in order to create a SharePoint counterpart.
- 4. Choose appropriate tools** – Both Microsoft and third parties offer tools to help with application migration.
- 5. Perform a proof of concept (POC)** – It is important to determine how applications in each category will be migrated. The POC should include at least one application from each of the categories you included in your inventory.
- 6. Perform the migrations** — The final step is to migrate most of your applications and rebuild the rest, especially the complex ones, in order to achieve the same—or perhaps even an improved—level of functionality.

Step 1: Inventory Your Notes Applications and Assign a Sponsor for Each

To begin, you need a complete inventory of all of the applications your organization has developed in Lotus Notes. Your Notes administrators should be able to produce this list fairly easily.

Then, for each application, identify an application sponsor who will be responsible for the following:

- **Intricate knowledge of the application’s purpose and functionality** – The application sponsor is often either the original developer of the application or a power user. However, with time, it often happens that the developer is no longer available and that notes about the context and functionality of the application are missing; this can make finding a sponsor with the appropriate level of knowledge more difficult.
- **Application justification** – The application sponsor is also responsible for justifying why the application is required in the organization. If an application is no longer required or if its functionality can be integrated into another application during migration, then it is one less application to convert, which reduces overall project costs.
- **The evolution of the application** – The application sponsor is responsible for requesting and justifying changes and updates to the application.
- **Application functionality testing** – The sponsor will also be responsible for testing the converted application to ensure that it operates properly and provides all of the features required by the organization.

Identifying application sponsors is an onerous task, but it is essential to the success of the project. Without an application sponsor to explain the application’s purpose, you still might not be able to determine whether the application is still needed. And without a sponsor’s in-depth knowledge, you might have to resort to expensive and extensive regression testing to understand an application’s full functionality. If you have to do this for every application, your project will become bogged down.

Step 2: Categorize Your Notes Applications

While your application sponsors are being identified, you can move forward with application categorization. Microsoft recommends four application categories:

- **Simple** applications can be mapped directly to existing SharePoint lists or site templates.
- **Medium** applications do not include workflows and can be mapped to customized SharePoint list or site templates.
- **Complex** applications include workflows and need both a custom list or site template and workflow development.
- **Highly complex** applications do not fit into the preceding categories and will require extensive development for migration.

Microsoft also lists a fifth category: non-SharePoint applications or applications that do not map to SharePoint functionality. In this case, the applications can still be hosted within a SharePoint environment, but their main functionality will be provided by other tools in the Microsoft infrastructure toolkit.

Most organizations find that a vast majority of their applications fit in the first three categories. Usually, less than 20 percent of the applications to be moved are highly complex, but these are the applications that will require most of your development efforts.

Step 3: Analyze Your Notes Applications

Next you need to analyze how each application works. This is usually a challenge, because many applications are likely to be undocumented and the original application developers gone. While your users may know the application and know what it does, they will not know how it does it. Therefore, the migration team will need to reverse-engineer the application in order to produce a workable substitute. Moreover, although many of your applications have taken years to develop into their current state, you will have only weeks to convert them into workable modules.

You need to identify the purpose of each application and its structure and level of customization, including the amount of workflow and other custom code contained within it. Specifically, you will likely need to do the following:

- Identify source and target fields to be mapped in the migration of the application
- Identify data external to the particular forms to be migrated (other views or databases) used in lookups
- Understand the business logic behind button-click events, other form/field events, and calculated fields

One key factor you will need to understand is access control lists (ACLs). Notes is a self-contained environment that provides its own security for all objects. SharePoint has SharePoint security groups, which are the functional equivalent of the Notes ACLs. A mapping must be performed between the security of the objects in Notes and the new SharePoint security objects.

Step 4: Choose Appropriate Tools

Although Microsoft does not offer any viable tools to help you execute and manage a migration from Notes, third-party vendors do. Investing in the right migration tool can significantly simplify the migration process and reduce cost by automating Notes database discovery, application classification, migration job setup, and provisioning of new SharePoint lists, libraries and sites.

Microsoft does offer development tools that can help you rebuild your Notes applications. These tools include the following:

- **SharePoint Designer** – Used to customize existing SharePoint templates and create new applications based on the building blocks included with SharePoint technology.
- **InfoPath Designer** – Supports the design of forms and their integration with SharePoint.
- **Visual Studio** – Generally used for applications that cannot be handled with SharePoint Designer or InfoPath Designer. Visual Studio is a powerful tool with a steep learning curve; it is discussed in more detail in the section “Rebuilding the ‘Tough Twenty’” later in this document.

While these Microsoft development tools can sometimes accomplish the bare minimum, a third-party workflow tool or a good set of third-party web parts can dramatically reduce the cost and time to migrate and rebuild your applications, as explained further in the section “Rebuilding the ‘Tough Twenty’” below.

Step 5: Perform a Proof of Concept

Once you have analyzed the applications to be migrated and chosen your tools, the next step is to perform a proof of concept (POC) to determine the migration cost for each application type

and gain buy-in from project detractors by showing them what can be done with each application type. The POC should take about one week.

Establish Partnerships

The best way to perform a POC is to partner with both provider organizations (for example, Microsoft) and third parties, such as consulting organizations that support migrations and vendors that provide solutions for migrating and converting applications. Your partnerships in the POC should include all of the groups that will eventually participate in your migration.

Create a Migration Test Bed

To prepare for the POC, you should create a fully-functional migration test bed. This environment should include SharePoint, Active Directory Domain Services, Exchange Server, Office Communications Server, and other target technologies. It should also include a linked Notes environment, which can be created by using either existing test or development environments or using a restored backup of a working environment.

To simplify the creation of the environment, you can rely on virtualization technologies and use virtual machines that simulate the capabilities of your source Notes environment. This will reduce costs, speed deployment and setup, and provide you with reusable components that can be repurposed once the tests are complete.

Select the Applications for the POC

While the environment is being prepared, another team can select a set of applications for the POC. The list should represent the range of applications in your environment, based on the categorization you made earlier.

For example, if you have a large number of complex applications but only a few simple, medium and highly complex applications, then your POC should include at least one medium, two complex, and one highly complex application. The logic is as follows: Converting a medium application is very similar to converting a simple application, so including a medium application obviates the need to include a simple one in the POC. Including two complex applications—the type you have the most of—lets you work with different kinds of complexities in this category and can help you prove you can overcome two or more migration challenges. Finally, demonstrating in the POC that you can migrate and rebuild at least one highly complex application will provide a lot of buy-in from project detractors. The section “Rebuilding the ‘Tough Twenty’” below explains how to tackle your most complex applications.

Microsoft Development Tools
Find the Microsoft tools here:
• [SharePoint Designer](#)
• [InfoPath Designer API](#)
• [Visual Studio](#) must be purchased independently.

Third-Party Migration Tools
Quest Software's Notes Migrator for SharePoint is a good option. Learn more at www.quest.com/notes-migrator-for-sharepoint.

What You Should Learn from the POC

The results of the POC should include the following:

- The cost of converting a single application of a given complexity.
- The time it takes to migrate an application of a given complexity.
- The tools required to perform a given migration.
- The skill sets required for the migration.
- The approach used to link the application sponsors, the Notes administrators, the partners you engaged and the POC staff together into a single team.
- The parameters used to perform both recursion testing and acceptance testing once the application migration is complete.
- These tests are vital for ensuring that data fidelity is retained after the migration.
- The reporting process used during the migration.

End-user Training
Another great advantage of the POC environment is that it can be extended to support end-user training once the Notes applications are converted.

Step 6: Perform the Migrations

Once you have completed the preparation steps above, for the “easy eighty” percent of your applications, the migration itself is nothing more than a manufacturing process of banging out each application in its new incarnation, testing it, and putting it into production.

However, you’ll have to rebuild some of your applications, especially the complex ones, in order to achieve the same—or perhaps even an improved—level of functionality. With the right tools, it can still be an efficient and consistent process that delivers a fully functional collaboration environment in SharePoint. The rest of the white paper details your options for rebuilding your most complex applications.

REBUILDING THE ‘TOUGH TWENTY’

Introduction

As we have seen, migrating most of your applications—which range from simple to complex—is a fairly straightforward process. But what about your highly complex applications? They usually comprise less than 20 percent of the applications to be moved, but these applications are often among the most critical for your business. If you can’t migrate or rebuild them to work in SharePoint, or doing so is too hard and too expensive, then your migration project will be left on the drawing board.

Using Native Tools

As we saw earlier, Microsoft offers several development tools to help you migrate your applications:

- **SharePoint Designer** – Used to customize existing SharePoint templates and create new applications based on the building blocks included with SharePoint technology.
- **InfoPath Designer** – An add-in application programming interface (API) that works with InfoPath. It supports the design of forms and their integration within SharePoint.
- **Visual Studio** – Generally used for applications that cannot be handled with SharePoint Designer or InfoPath Designer.

Unfortunately, chances are good that your organization will have few, if any, staff proficient in these tools. In fact, these tools will likely be completely new to your staff. SharePoint Designer and InfoPath Designer are fairly easy to learn and can help you reconfigure many of your simpler applications. But for your most complex applications, using native tools means using Visual Studio.

Visual Studio is a powerful tool, but it requires extensive training, so you may have to hire experts to support your migration. Expert developers are expensive, and their involvement usually adds a lot of time to the project, in analysis, design, testing and deployment. These added costs and time requirements may keep you from being able to migrate your most important applications—and thereby might force the organization to abandon the migration project altogether.

For example, organizations often hire expert developers to create development templates in Visual Studio, and then have their in-house, novice developers use those templates to create new code; the template guides the novices during the development and helps them create complex code without having to be experts themselves. But it takes time—often one to two months—to develop custom templates that meet your particular requirements, so you may find that nothing seems to be happening in your project as you wait for the templates to be completed. These delays can make it difficult to get buy-in from the project stakeholders, and if the cost of migrating is too high, then there may be no value in it at all.

There is, however, another way.

Rapidly Rebuild Notes Applications with Third-Party Tools

Ten years ago, organizations migrating from Notes had little choice but to spend the time and money required to use Visual Studio to rebuild their complex applications in SharePoint. Today, however, SharePoint is a mature platform, and many third-party tools are available to support migrations. These solutions often include premade components that can more easily transport and reconfigure Notes feature sets into a SharePoint environment up to 80 percent faster than custom coding. These premade components include both web parts and ready-made application templates.

Reduce Development Time and Cost

Third-party SharePoint web parts and templates can dramatically reduce the time, cost and development resources required to rebuild complex Notes applications in SharePoint. Specifically, they can:

- **Reduce the time required to rebuild a complex Notes application up to 80 percent** – By providing canned actions, formulas, and Notes-like application templates that can be configured with the click of a mouse rather than recoded from scratch, web parts and templates from third parties can reduce the migration timeframe from months to a matter of days.
- **Provide a familiar development experience** – Third-party tools provide an environment similar to what your Notes developers already use, which will ease buy-in from your Notes developers and help them want to be part of the project instead of resisting change.
- **Reduce training costs** – Little or no training is required to use these web parts and templates. They usually offer intuitive configuration and computer-based training to help even non-developers and IT people get up to speed quickly, in a matter of days. In contrast, tools like Visual Studio can require extensive and expensive training.
- **Reduce developer costs** – Your organization may be able to avoid hiring migration or development experts and instead rely more heavily on internal staff.
- **Require less SharePoint expertise** – Third-party web parts can be configured without in-depth SharePoint experience.
- **Offer strong ROI** – Third-party tools may even provide so many Notes-like components that you may find that you do not need to rebuild applications outside of SharePoint.

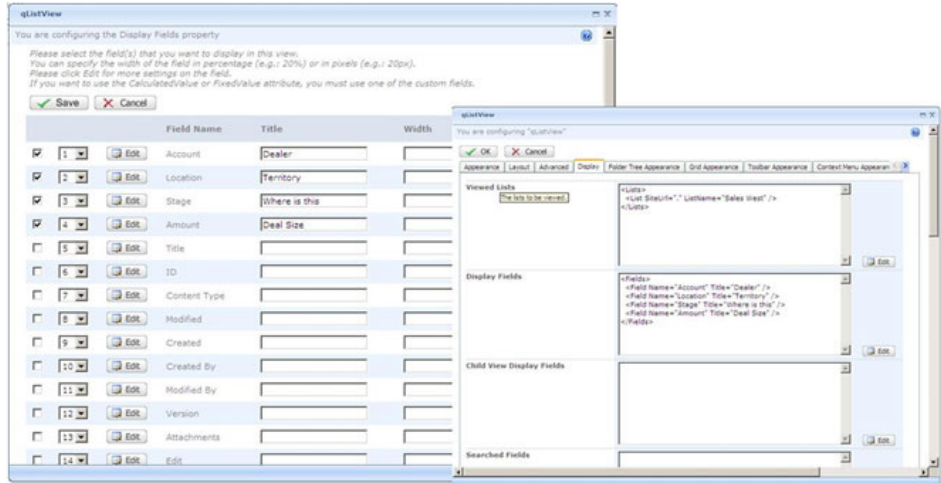


Figure 2. Quest Web Parts enables you to configure rather than code, so you can rebuild your Notes applications quickly and easily in SharePoint.

Third-party Web Parts Quest Software has been building custom SharePoint web parts for some time. Learn more at <http://www.quest.com/web-parts-for-sharepoint>.

Example: Configure instead of Code with Quest Web Parts

Exactly how can a third-party tool save you so much time? With Quest Web Parts, for example, you use an intuitive configuration interface to rebuild your applications quickly—with no coding required. (You can also extend the product with coded, custom actions if needed).

Doesn't that look much easier than using Visual Studio?

Deliver a "Notes-like" Experience for your Application Users

Saving time and money is nice, of course, but equally important is delivering a SharePoint environment that has the functionality your users need and a look and feel that they will accept.

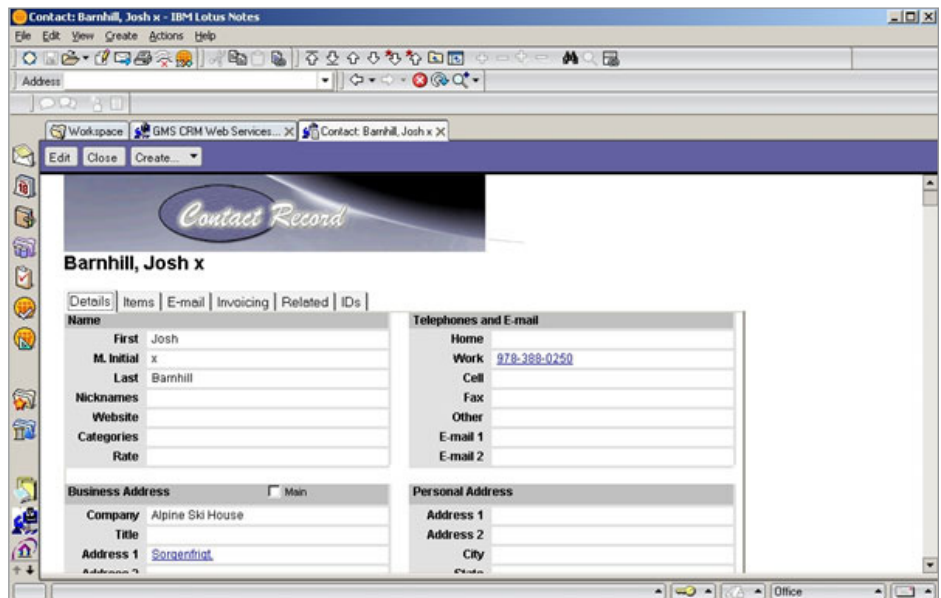


Figure 3. A CRM application in Lotus Notes

Third-party tools can help you replicate a “Notes-like” experience for your application users. This can ease the buy-in from end users who are resisting adoption of SharePoint, and help users become productive in the new environment faster.

For example, consider the following customer relationship management (CRM) application in Lotus Notes (see **Figure 3**).

Here’s that same application, rebuilt in SharePoint using Quest Web Parts. Note the similar look and feel (see **Figure 4**).

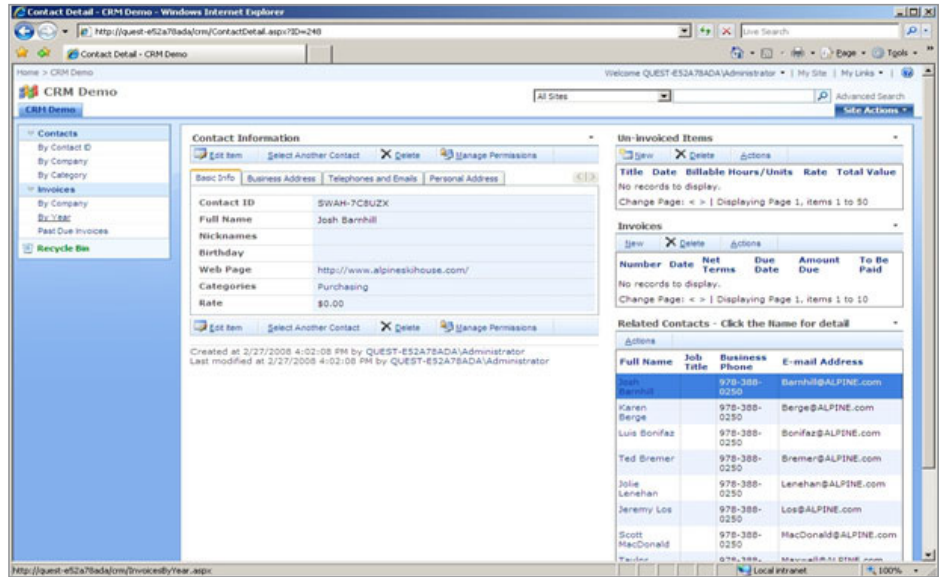


Figure 4. The same CRM application rebuilt in SharePoint using Quest Web Parts

Deliver Functionality not Available with Native SharePoint

In addition, third-party tools also enable you to deliver features that simply aren’t available in native SharePoint:

- Enhanced data viewing capabilities, such as list groupings, calendar rollups and compelling 3-D charts. For example, many Notes applications have multiple calendars with combined views; you can’t do that in native SharePoint, but you can with third-party tools.
- Dynamic data relationships and business logic, such as hide-when/show-when on forms and parent-child relationships between SharePoint lists that are enforced for referential integrity.
- Security-trimmed tabbed forms.
- Rich discussions.
- Data surfacing tools that simplify the integration of Notes data into SharePoint.

For example, consider discussions. Lotus Notes discussion boards are intuitive and user friendly; native SharePoint discussion threads are not. This can make the transition to SharePoint more difficult for users. Quest Web Parts helps remove this hurdle by enabling the creation of easier-to-follow, multi-threaded discussions, which are much more familiar to a Notes user.

Consider this discussion board in Lotus Notes (see **Figure 5**).

Here is the same discussion board in SharePoint, rebuilt using Quest Web Parts (see **Figure 6**).

While native SharePoint only supports two levels of row grouping in its list view, notice in **Figure 6** that the Quest Web Part discussion list can be configured to have the same number of grouped levels that are used in the Notes application along with a familiar document content preview panel. Building a discussion board with this Notes-like look and feel is not possible using native SharePoint, but it is easy with Quest Web Parts—and it will make your users happier and more productive.

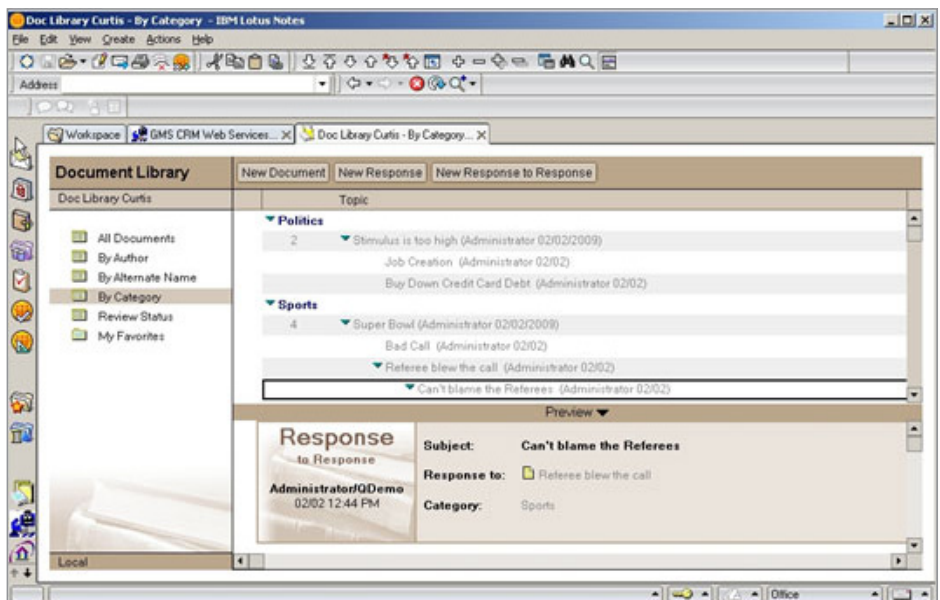


Figure 5. A discussion board in Lotus Notes

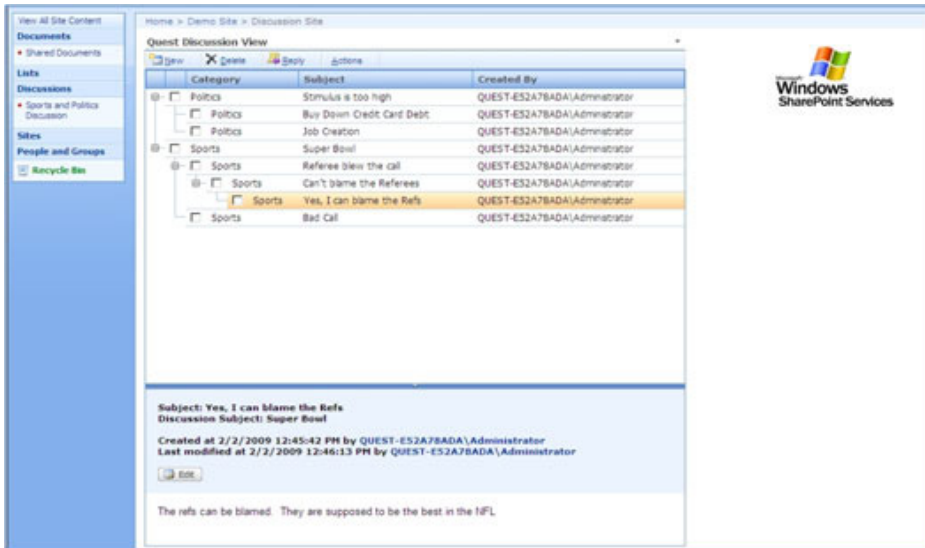


Figure 6. The same discussion board in SharePoint, rebuilt with Quest Web Parts

CONCLUSION

Success is defined by results. Take the time to evaluate third-party web parts and templates for your Notes-to-SharePoint migration. By enabling you to rapidly rebuild complex Notes applications in SharePoint with minimal development resources, these tools can ensure a faster, less expensive migration and a resulting SharePoint environment that serves your end users well. Those are a sure sign of success for the migration project.

Prior to joining Quest, Curtis worked as an executive director and vice president of IT at Cox Communications (the third largest cable operator in the U.S.), as a consultant for IBM and American Management Systems (AMS) and as a programmer/analyst at Sun Microsystems.

Curtis earned an MBA from the Wharton School of Business at the University of Pennsylvania and a BSBA in Management Information Systems from the University of Nebraska.

ABOUT THE AUTHOR

Curtis Kelly is the product manager for Quest Web Parts for SharePoint. He has more than 20 years of experience in the IT industry and has held various executive, management and programming positions in application development and IT consulting. Curtis is skilled in SharePoint integrations and configurations and the Quest SharePoint product line. His expertise includes architecting operational support systems, workflow and automation using Java and/or .NET technologies.

Prior to joining Quest, Curtis worked as an executive director and vice

president of IT at Cox Communications (the third largest cable operator in the U.S.), as a consultant for IBM and American Management Systems (AMS) and as a programmer/analyst at Sun Microsystems.

Curtis earned an MBA from the Wharton School of Business at the University of Pennsylvania and a BSBA in Management Information Systems from the University of Nebraska.

Sponsored by

