A Lotus Software White Paper
August 2002

Lotus® software

Technical Strategy and Domino Developers' Roadmap

THE INFORMATION CONTAINED IN THIS DOCUMENTATION IS PROVIDED FOR INFORMATIONAL PURPOSES ONLY. WHILE EFFORTS WERE MADE TO VERIFY THE COMPLETENESS AND ACCURACY OF THE INFORMATION CONTAINED IN THIS DOCUMENTATION, IT IS PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. IN ADDITION, THIS INFORMATION IS BASED ON IBM’S CURRENT PRODUCT PLANS AND STRATEGY, WHICH ARE SUBJECT TO CHANGE BY IBM WITHOUT NOTICE. IBM SHALL NOT BE RESPONSIBLE FOR ANY DAMAGES ARISING OUT OF THE USE OF, OR OTHERWISE RELATED TO, THIS DOCUMENTATION OR ANY OTHER DOCUMENTATION. NOTHING CONTAINED IN THIS DOCUMENTATION IS INTENDED TO, NOR SHALL HAVE THE EFFECT OF, CREATING ANY WARRANTIES OR REPRESENTATIONS FROM IBM (OR ITS SUPPLIERS OR LICENSORS), OR ALTERING THE TERMS AND CONDITIONS OF THE APPLICABLE LICENSE AGREEMENT GOVERNING THE USE OF IBM SOFTWARE.

Contents

Abstract

This paper articulates IBM®'s vision for the future of collaboration and the extent to which current Domino™ applications and developer skills can be reused and leveraged in the next generation of collaborative applications. It explains IBM's plans to continue investment in Lotus® Notes® and Domino™ as a complementary and parallel architecture to IBM WebSphere®, affirming Lotus' leadership in collaboration at IBM and throughout the industry. IBM has no current plans to merge Lotus Domino with WebSphere or to require WebSphere to run Domino applications in the foreseeable future.

This paper also helps you understand the core strengths and the value of both Lotus Domino and WebSphere. It describes a set of application criteria and gives guidelines for using either Lotus Domino or WebSphere or a combination of both to most successfully meet those criteria. This paper concludes with a call to action to help you prepare for supporting and deploying integrated collaborative solutions in the future.

The Lotus Technology Strategy and Developers' Superhighway

Early in 2002, Lotus Software announced its technology strategy for the next generation ("next gen") of collaboration. The goal of the next generation strategy is to "unleash" the collaborative capabilities of Lotus products to enable them to be more easily integrated with each other and with custom or third-party business applications. Lotus' vision is to:

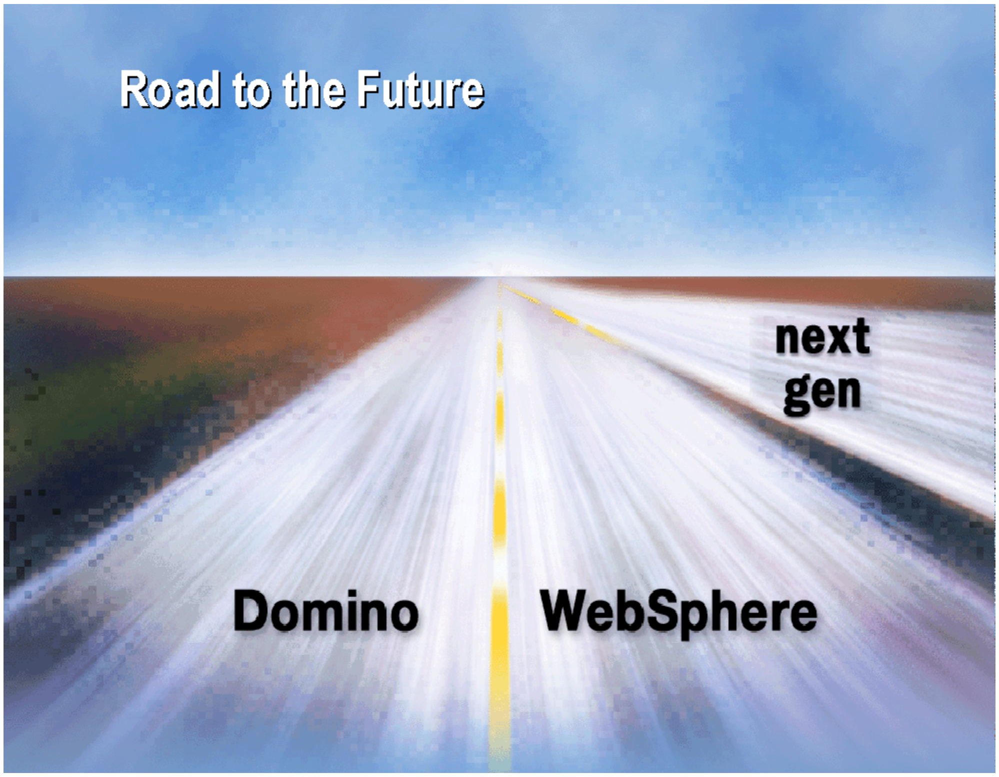
- Allow collaboration to spread into more areas of the business
- Further enhance employee productivity
- Enable widespread teams to work together efficiently in context of the business process

IBM believes that a new work-paradigm called "contextual collaboration" will emerge in which employees collaborate with each other "in context" of business processes. Instead of using separate, stand-alone tools for collaboration, users will have features for collaboration at their fingertips wherever they work — be it in a portal application, a Windows application, a Web application, or a mobile device.

To implement this vision, IBM Lotus will deliver modular collaborative services for easier cross-Lotus-product integration, and easier integration of Lotus products with third-party and custom applications. The technical implementation is a collection of services-oriented interfaces for all Lotus server products, including Lotus Domino. Today, Lotus software APIs differ widely from product to product, making integration rather complex. Lotus software will simplify integration by introducing a common cross-product schema and data model to make the APIs more uniform among products. These APIs will make it easier for your organization to build cross-product integrated applications. Lotus will initially produce Java™ interfaces, but will move to Web services as those technologies mature.

Your existing custom applications built with Lotus products will not be affected, allowing further leveraging of your application investments. In the future, your applications can take advantage of the next gen collaborative services. For example, Domino Inbox services or Calendaring services will be exposed for integration into any application that can use Web services, so your developers can easily build these features into many different types of business applications.

To fully understand the strategy and how it applies to Domino applications and their development, envision a multi-lane superhighway with a Lotus Domino lane, a WebSphere lane, and a next gen lane. The Lotus Domino lane continues as far as the eye can see and runs parallel to the WebSphere lane, which also continues as far as the eye can see. The next gen lane merges with the other two lanes at a point in the future.



In the Lotus Domino lane, the Domino collaborative application development and deployment environment enables you to develop applications quickly and to take them off-line, bringing people, processes, and data together to facilitate both productivity in e-business and quick decision-making. Lotus will continue the current Domino application development model and data store (NSF) and in the future, will enhance it to meet customer and developer requirements. As in the IBM tradition, Lotus Notes and Domino customers will benefit from comprehensive support for the foreseeable future. Additionally, Lotus Domino will increase its support of the Java 2 Platform, Enterprise Edition (J2EE) and infrastructure standards, such as Java Server Page (JSP) tags, Java APIs, LDAP, and RDB integration to assist developers interested in working in both the Domino and WebSphere lanes of the superhighway.

In the WebSphere lane, the J2EE specification is leveraged as the application development platform. J2EE provides a specific architecture for building, deploying, and managing applications in multiple tiers, often broken into presentation, logic, and data. This architecture is designed to provide scalability, flexibility, and manageability. While J2EE is a rich application development platform, it has very few features to support collaboration, so it benefits from having Lotus Domino to provide rich collaborative capabilities. Applications designed to use the Lotus Domino and WebSphere lanes blend powerful collaborative features with significant transactional scalability to deliver end-to-end e-business solutions.

In the near future, the third lane of next gen collaborative services will intersect with Lotus Domino and WebSphere. The next gen lane will provide both Lotus Domino and WebSphere developer communities with a modular architecture for building integrated collaborative solutions. Domino developers can leverage technologies in both the WebSphere and next gen lanes to augment and to enhance their applications. WebSphere and J2EE developers can integrate collaborative capabilities from Lotus Domino and other Lotus products through Web services. In both cases, integrating collaborative solutions becomes easier and can be done with standard developer skills.

Lotus' role is to develop software that not only makes new things possible, but also makes them easier and cheaper. That's what this strategy is all about — a new architecture for the next generation of collaboration, to allow collaborative solutions to be integrated virtually anywhere in the enterprise to boost productivity and communication in all areas of your business.

The Value of Domino Applications and Application Development Model

The value of Domino applications

Lotus Domino is a comprehensive application platform for collaboration that handles both connected and disconnected requirements for data and applications. Most customers initially purchase Lotus Domino for the built-in enterprise e-mail and calendar/scheduling, making those types of applications the most widely deployed collaborative applications. However, most customers exploit the “more than mail” capabilities that support core business processes which enable employees to work together efficiently and securely. Lotus Domino is comprehensive because it provides the complete infrastructure needed to create, test, deploy, and manage distributed, multi-lingual applications including directory, database, application server, administration, security, connectivity, Web server, e-mail server, calendaring engine, and so on — all in the box.

Domino developers can design applications for the Lotus Notes client, Web browsers, mobile phone and handheld devices, or most commonly, for a hybrid environment accessed by multiple types of clients. Hybrid client Domino applications can leverage replication and off-line services, for secure synchronized applications that work as well in a disconnected mode as when accessed on a server over a network. Replication enables users to save a local copy of a Domino application and its data on a file system and to periodically synchronize the data, so users can be productive and efficient even when disconnected from the network. If Domino Off-Line Services (DOLS) is added to a browser client, then users can work with Domino Web applications in a disconnected state with similar productivity results.

Examples of Domino solutions include document-centric and workflow process routing, such as project teamrooms, document repositories, discussion forums, sales force enablement, and employee self-service applications. Businesses of all sizes have benefited from Domino applications.

The United States Navy uses Lotus Domino to securely collaborate between ships in battle groups. Implemented in only a couple of months, the Navy’s “Collaboration @ Sea” project leverages advantages of Web browser accessibility as well as replicated and distributed data stores to keep all participants synchronized while meeting bandwidth challenges.

Domino applications were also built in a matter of hours following the tragic events of 9/11 for the City of New York to manage a distributed repository of the status of structures and of missing people around Ground Zero. The open platform of Lotus Domino helped enable the data collected to be shared across many assistance agencies. Lotus Domino is a mature and widely deployed application development and deployment product with the next release to be its sixth.

The value of Domino application development

As a comprehensive application platform, Lotus Domino includes a tool for rapid application development (RAD), a document-based object model and broad programming language support for building custom collaborative applications. With these choices, your organization can leverage many developer skills to develop a Domino application. For example, with minimal programming background and a need for a collaborative application, such as a discussion forum or document library, you can easily create a Domino application from a ready-made template without writing any code.

If you want to add workflow to move documents around in a role-based workflow, you can add macros and formulas to the original template. When conditional logic needs to be added to send off notifications based on the contents of a document or view of data, a developer with basic programming skills can enhance the application further. In a typical custom Domino application, very basic programming skills are sufficient for the majority of tasks.

For more advanced solutions, developers can use Java, Microsoft® COM, C/C++, or CORBA. The multiple interfaces of Lotus Domino to a single object model enable a developer to pick the best language for the task, reusing his skills in new applications and solutions.

Some solutions require non-Notes data and multiple language or globalization support. The rapid application development capabilities of Lotus Domino facilitate such solutions. Using visual data mapping techniques, a developer can easily, quickly integrate relational data with Notes data. No programming is required with technologies known as Lotus Connectors. Domino applications that require global deployment can have all elements translated to a variety of languages with some straightforward forms. All of these RAD activities take place in the integrated development environment called Domino Designer.

The features of Domino application development in Domino 6 can benefit your organization by increasing code and design element reuse, by simplifying team development, and by providing general enhancements and improvements over Release 5, all of which will help improve developer productivity. Major new features include:

- XML management (DOM parser, SAX parser, XSLT transformer, XML import and export utilities)
- Programmatic rich text handling
- JSP tag interface
- Team development features, like design element locking
- Tighter integration with external systems, such as relational databases
- Dynamic Web page creation tools

As you can see, Lotus will continue to enhance the Lotus Notes and Domino application model with new features as required by customers. In coming releases of Lotus Notes and Domino, you can expect Lotus to stay consistent with its open standards platform strategy by adding functions for mobile/wireless application development, by making consumption of Web services easier, and by enhancing XML.

The Domino roadmap for application development builds on the fundamental premise that Lotus Domino is a flexible and open platform demonstrated by XML and broad programming language support. Flexibility and openness are key to a Domino application's ability to leverage J2EE, straddling over to another lane on the previously-mentioned application superhighway. You can extend your Domino application investment in data and application logic, exposing relevant applications and data using Web services (through LotusScript or Java) or JSP tags to integrate with next gen or WebSphere applications.

Along with the progressive innovation that Lotus will add to Domino Designer and to the Domino programming model, there will be continued integration with WebSphere Studio. This integration will not replace Domino Designer, but will help facilitate teams of developers using both Domino and WebSphere Application Server in their environment. Such teams will benefit by utilizing the strengths of each system when building applications. For additional information about future application development and integration points between Domino Designer and WebSphere Studio, see the section "The Future of Rapid Application Development" later in this paper.

Domino applications and Web services

Web services is a collection of emerging standards that simplifies application integration by providing a standardized access protocol known as Simple Object Access Protocol or SOAP. Developers can express the application interface using XML. Today, Domino developers can add Web services interfaces to existing Domino Release 5 applications on any platform, using about 35 lines of LotusScript® (BASIC) or Java code. The only tools required to use Web services in your Domino applications are a Lotus Domino server and Domino Designer. Samples are available at http://advisor.com/Articles.nsf/aid/DEVEG02, and you can download a toolkit from http://www.alphaworks.ibm.com.

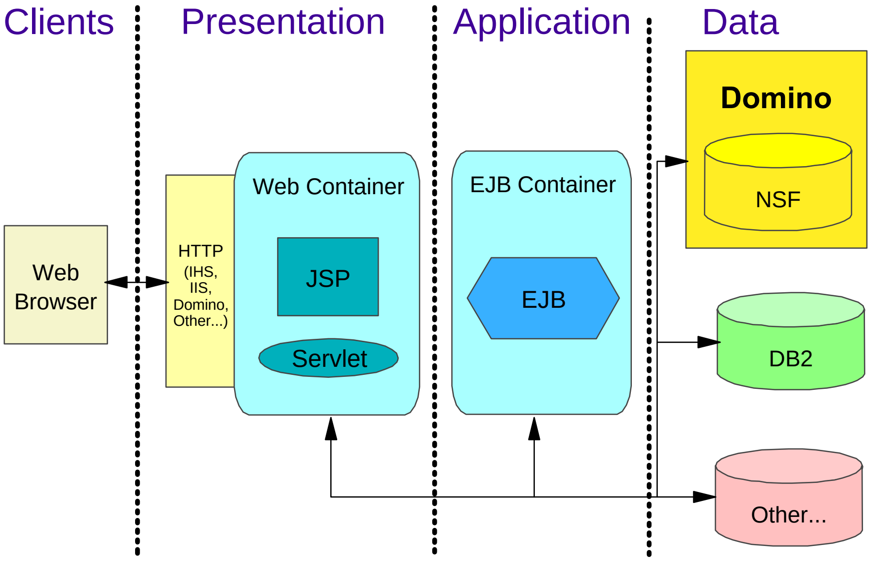
Lotus Domino can host Java-based Web services that expose Domino data and functionality. Using a combination of a J2EE server like WebSphere, appropriate SOAP classes, and the Domino Java Objects, developers can expose desired portions of their current Domino applications as Web services. To do so requires some Java development skills and knowledge of the Domino object model. IBM's WebSphere Application Server ships with the needed SOAP classes, and the WebSphere Studio development environment has wizards for creating, consuming, managing, and deploying Web services. This helps improve developer productivity and lets developers focus on exposing the essential parts of a Domino application as the Web services.

The Value of J2EE Applications and Application Development Model

Traditionally, J2EE applications use a tiered application model. The key elements of an application are divided into distinct components that perform their specific functions in an environment separate from the other elements. The simplest tiered model is broken up into presentation logic, application logic, and data housing. Each of these tiers performs a specific function of the application and can reside on one or more systems. An enterprise can deploy application elements to any type of hardware and location for the maximum benefit of the business.

This tiered model provides valuable flexibility when building, deploying, and reusing applications. Based on this type of architecture, an application's presentation logic tier can be located geographically near the users on inexpensive hardware; with the application logic tier located at a corporate headquarters on a more powerful, expensive system; and the data housing tier on a mainframe system that has been running the business for decades. This example is quite extreme, but possible using the J2EE tiered application model. Another benefit of this model allows an application's elements to be easily reused to benefit some other part of the business.

As an example to consider, an inventory application built around this model that lets manufacturing know how many widgets are in the warehouse and how many should be made based on sales forecasts. The sales force also needs an application to know how many widgets are in the warehouse and what the production schedule is for manufacturing, so it can set customer expectations for filling orders. Instead of the new sales application duplicating the functionality of the manufacturing application, the manufacturing applications logic can be reused and tied into the new sales application.



J2EE application servers

As organizations invest in a J2EE server platform, choosing the right one for their environments can seem a daunting task. Yet, while there may be many choices, the core functionality of all J2EE servers remains more or less the same. These core elements of the J2EE application server model are put forth by the J2EE specification and provide a vast array of services to be provided to and handled for the application that runs on it. The server handles the life cycle of the various application elements, providing them with current state and context information, authentication information about the current user, transactional context for highly sensitive applications, and so on.

It’s where the J2EE specification leaves off, that a J2EE server vendor can provide enhancements that make one server more appealing or appropriate to a certain customer’s environment. This gives server vendors the ability to develop value-added functionality without impacting the standards to which the servers must comply. Therefore, the development model for each server from various vendors is similar and allows customers the ability to leverage the skills of administrators and developers from any one vendor’s J2EE server to another vendor’s server.

Domino and J2EE

One of the key differences between the J2EE application server and the Domino application server is that Lotus Domino provides a fully integrated environment. It handles nearly all aspects of the application environment from application execution to user authentication, directory services, data hosting, and presentation display all in one system.

The J2EE model differs in that it has elements for providing the application with all the same information, but the J2EE server is not responsible for fulfilling all the aspects that the Domino server fulfills. The J2EE server calls out to different parts of the customer environment to fulfill the requests of data, directory information, and so on. For example, while the J2EE specification outlines how the server can get data from a data store into the application using JDBC, J2EE does not require that the server contain the database manager itself.

Domino features enable applications to integrate with J2EE

You can reap many application benefits by combining Domino and J2EE applications, most importantly the addition of human interaction into scalable and transactional applications built on J2EE. When integrating Lotus Domino with a J2EE-based application, there are two key ways to access Lotus Domino:

- Domino Objects for Java
- Domino 6 custom JSP tags

The Domino Objects for Java are essentially the Domino Java API. These objects allow for object-based access into the data store for the key database procedures: create/read/update/delete, better known as CRUD activities. This API allows access not only to Domino data but also to some key services, such as registering users, running agents, sending mail, and others.

The Domino 6 custom JSP tags are for use only from within a JavaServer Page. JSP tags are XML tags embedded in a JSP providing data access, data input, and process control. These tags abstract the Domino Objects for Java and provide a quick development turnaround for building a J2EE application that uses Domino data and services.

Lotus Domino also has extensive XML capabilities that can be used from a J2EE application. A J2EE application can use the XML interface to Domino to access the unstructured Domino documents without knowing specifics about the data stored in Lotus Domino. Then, the J2EE application can process the resulting XML and transform it to the specific format needed by the application. When the application is ready to update Lotus Domino, the same XML interface can be reused to update the database through XML, thus providing for a round-trip of data in and out of Lotus Domino.

Domino integration with WebSphere Application Server

Within IBM, the Domino and WebSphere Application Server teams have coordinated efforts to develop and to deliver key integration points that customers have requested. Each of these elements stem from the need for excellent user experience, easier administration, and overall infrastructure integration. While Java and J2EE standards make it possible to integrate disparate applications at a fairly broad level, Lotus and WebSphere have developed powerful integration points that are only available between Domino and WebSphere Application Server. For example:

- Many customers have chosen a strategic HTTP server for their environment. Lotus Domino has a built-in HTTP server that many customers use, but customers also want Lotus Domino to integrate better into their infrastructure. Coming in Lotus Domino 6 is the ability to use a third-party HTTP server in front of Lotus Domino as the main point of entry for Web browser users. This functionality is provided by a set of HTTP plug-ins that were developed by the WebSphere Application Server team and that allow the use of all major HTTP servers. When Lotus Domino 6 is made generally available, there will be initial support for Microsoft Internet Information Server (IIS) on Windows NT® 4.0 SP6 and on Windows® 2000 Advanced Server as well as IBM HTTP Server (IHS) on AIX. Support for additional HTTP servers, including Apache, iPlanet, and Domino, on various operating systems, will be added in future maintenance releases.

- For customers looking to standardize on a single user directory, Lotus Domino can serve the Domino Directory to other applications using Lightweight Directory Access Protocol (LDAP). LDAP is quickly becoming the defacto standard for directory serving and is also used by WebSphere Application Server for authenticating users. Because Lotus Domino can serve as an LDAP server, WebSphere can utilize this directory, allowing customers to manage user identities for Lotus Domino and for WebSphere in one location, easing administration skills needed in customer environments.

- In cases where WebSphere Application Server and Lotus Domino use the same LDAP directory (not limited to the LDAP capabilities of Domino alone), customers can configure Single Sign-On (SSO) between Lotus Domino and WebSphere. SSO allows a browser user to enter his authentication information once and to have that information stored in a secure context that both WebSphere and Lotus Domino share. With SSO enabled, users navigating between WebSphere and Domino applications do not need to authenticate multiple times as they would without SSO enabled.
- In addition, the Domino Objects for Java (the Domino Java API) allow for the secure processing of data in Lotus Domino by allowing the use of the SSO token to create a session in Lotus Domino. Programmatically, a developer can use the SSO token to create a secure session with Lotus Domino as the current Web application user from WebSphere Application Server. This eliminates the need for an application to know the specific user name and password when accessing Lotus Domino and also minimizes a security risk of users' passwords being handled by any given application.

Choosing a Lane

This section is intended to be used as an assessment tool to help you understand the factors that should be considered when planning for integrated solutions on IBM server platforms. The section is divided into subsections representing functional areas. Within each area are questions that may help you understand the factors that can determine the most suitable architecture for the solution — in other words, which parts of the application should be designed for which "lane."

The way to use this section is to have an application (either current or planned) in mind and to answer each question as it applies to that application.

1. Application purpose and use

Application content

Will the application include rich format content editing, creating, and rendering?

If a large part of the application's use includes creating/writing/updating/deleting richly formatted text or if the application needs to display, store, and process Microsoft Office documents, then this application fits squarely in the Domino lane. Additionally, the application is probably best suited for the Lotus Notes client because of its built-in rich text editor and Microsoft Office integration.

Transaction processing

Is the application performing transactional tasks?

A transactional application is one in which a series of steps in the application is monitored so that the process can be completely reversed if necessary. A classic example is a banking system in which an account credit must be reversed if the preceding debit step isn't complete. Transaction processing fits squarely in the WebSphere lane, with Domino applications providing the collaborative services for the human intervention necessary for resolving exceptions that occur in business transactions.

User-centric forms-driven workflow

Does your application need to route electronic forms to users in a series of predefined steps to complete a workflow process?

An example is an employee self-serve help desk application in which the employee fills out a form that when submitted, is routed by e-mail to the appropriate help desk professional. Upon closure of the problem, the completed form is routed back to the originator for sign-off. After sign-off, the form is logged into a help desk tracking database. These types of solutions are relatively easy to build and to deploy on Lotus Domino due to the built-in e-mail APIs and flexible roles-based identities and access control. A value-added solution is to integrate Lotus Domino and WebSphere plus IBM MQSeries™, using Lotus Domino for the user-centric forms-based workflow that occurs in support of transaction-processing workflows that are most suitable to be managed by WebSphere plus MQSeries. Such end-to-end workflow solutions that marry systems transactions with human processes is a capability that no other single vendor can offer.

Disconnected (off-line use)

Does the application need to be used from a computer that is disconnected from the network with full application fidelity?

Application fidelity means that the application functions the same, whether on-line or off-line, including creating/editing/updating/deleting data. This is one of the unique features of Lotus Domino. Only through Domino replication can an application be taken off-line and used with full fidelity and access to data from a Web browser or Lotus Notes client. When the user reconnects, the work he/she did off-line is fully synchronized with the server. This is another feature that is available only in Lotus Domino.

Notes client access

Lotus Notes provides a desktop environment for maximum user proficiency in collaborative applications, such as e-mail, calendar/scheduling, personal information management, and custom collaborative applications. Lotus Notes provides a graphical window user interface and an extensible client framework for running collaborative and document-centric applications. The following list includes Lotus Notes client features that differentiate it from a Web browser:

- Richly formatted content editing (tables, spell check, in-line graphics, full font support, find/replace, etc.)
- Microsoft Office integration for attaching, storing, and distributing files
- Off-line/disconnected e-mail, calendaring, PIM, and applications with virtually full functional fidelity while off-line and synchronization when reconnected
- Data encryption with security features, wire encryption, and digital signatures not available in a browser environment

Access control

Do users need to delegate their actions?

Some applications require a feature that grants one user the access control rights of another user. This is most frequently used to enable one user to act on behalf of another, while tracking which user did what. A classic example is a manager delegating responsibilities to an assistant. The assistant should not receive equal access rights to *be* the manager, but should get access rights to *act on behalf of* the manager. The difference is that any create/write/update/delete actions by the assistant are noted separately from those actions by the manager for auditing purposes. This feature depends upon native RDB access control and should therefore live in the WebSphere lane.

Does a user’s access control need to be limited to a subset of records in a table or fields on a form?

For example, a manager is granted access to the “employee salary” table, but he/she can only see the records of the employees that he/she manages. Likewise, a user filling out a form might see only certain fields, while a manager using the same form has an expanded instance with additional fields shown. Lotus Domino has a unique security model in which access control can be applied to very granular subsets of the application and the data. Access control can also be altered programmatically on the fly, adjusting who sees what based upon a series of dynamic conditions. Therefore, this feature belongs in the Lotus Domino lane, but would add unique value to WebSphere applications.

Does the application need dynamic roles-based user identities?

Most applications make use of Groups in the directory. However, there are many cases in which a group is application-specific. Storing application-specific groups in a corporate directory is unwise because of management implications, so a better solution is to incorporate the user roles into the application, storing the roles as part of the application data, and writing logic to determine the role of the current user.

A good example of an application that requires roles is an approval process. For example, in an HR hiring application, the people who need to approve a salary offer are the HR manager, the finance manager, and the hiring manager. Each of them is a member of different groups in the directory, but salary approval rights cannot be granted to these groups because they contain too many other people. A new group called Salary Approval cannot be created because the hiring manager member changes with every hiring instance. So the solution is to create an application-specific role that users move in and out of.

Role membership is changed on-the-fly through application logic, and administration is done at the application level instead of at the directory (systems) level to simplify application administration overhead. Any portion of an application that requires this type of application-specific and/or dynamically altered roles fit in the Lotus Domino lane. This is another advantage of integrating Lotus Domino with WebSphere applications because many HR solutions are J2EE-based.

2. Scalability requirements

Number of users

Will the number of concurrent Web browser users exceed 5000?

There are several factors that determine the scalability requirements of a Web application, one of which is the number of concurrent, active users of the application. As a general rule, if the number of concurrent active users exceeds 5000, and if the application is deployed to Web browsers as opposed to Lotus Notes clients, then the application is probably better suited for the WebSphere lane.

However, if the application is deployed to Lotus Notes clients, then choose Lotus Domino. Lotus Notes uses a client/server architecture in which much of the processing is done client-side to permit greater server scalabilty.

Infrastructure configuration for scalability

Does your application need to scale horizontally and/or vertically?

Deploying the appropriate hardware configuration is key to meeting application scalability and availability requirements. Horizontal scalability refers to distributing an application across several physical servers or several server instances in a “blade” configuration. Vertical scalability refers to upgrading to a more scalable hardware/operating system (HW/OS) platform. Be sure to consider both the scalability requirements for today and for the future. Both WebSphere and Lotus Domino take advantage of vertical scalability because both servers support a broad range of HW/OS platforms from Windows servers at the low end to the IBM e Server z-Series at the high end.

WebSphere has the distinct ability to scale horizontally. One of the key features of WebSphere is the ability to configure applications to run distributed across multiple servers, which adjust the load balance dynamically to meet variable runtime conditions.

3. Data requirements

Data model

Does the application need a structured or unstructured data model?

There are several types of data storage: relational and hierarchical being the two that apply to this paper. Relational storage maintains multi-dimensional links between “related” records. Hierarchical storage uses a comparatively flat parent/child relationship model. Typically, relational storage is better suited for structured data where “many-to-many” relationships exist between records, such as the linkage between customer records, invoices, purchase orders, and vendor records.

Hierarchical storage is better suited for unstructured data, such as documents, lists, Powerpoint presentations, and other forms of “free form” information. Another characteristic of hierarchical data is the ability to change the data structure, such as adding or deleting fields at will without rebuilding the database. Other characteristics unique to unstructured storage are:

- Allows a field to have multiple values
- Allows simultaneous editing of records by different applications or clients (no record locking)
- Works best for relatively simple (low complexity) searches and searches of document contents

- Works best with data that lends itself to being displayed as a categorized list or in a single table

Many applications require both relational and hierarchical data storage, so this is perhaps the strongest case for building applications that straddle the Lotus Domino and WebSphere lanes. Lotus Domino has a built-in unstructured data store and WebSphere has robust connectivity to RDB systems.

Does the application need direct access to a system of record?

A system of record is a relational database feature that enforces existence of one and only one instance of a record. This is a requirement of many applications that run in regulated industries, such as insurance, health care, and pharmaceutical. This feature fits squarely in the WebSphere lane because Domino replication is counter to a system of record requirements.

Data volume

Will the data volume exceed 2GB?

The amount of data storage needed for an application is an important factor in determining its architecture. Other data-related factors also apply, such as the degree to which users interact with the data with create/write/update/delete (CRUD) activities and search requirements. Since Domino Release 5, there is no longer a hard limit on the size of a Domino database, but managing high data volume and high CRUD activities is better suited for the WebSphere lane.

However, a common exception is the need for storing unstructured data, such as documents, and for supporting rich text/highly formatted content, which is better suited for the Lotus Domino lane. The ideal solution in this case is to build an integrated solution. If it is necessary to store the application data in a single database, then the Lotus Enterprise Integrator (LEI) can synchronize the Domino database with the RDB.

Data distribution

Does the application data (exclusive of the app logic) need to be distributed to servers across a WAN or the Internet?

In many cases, an application is architected for execution across distributed servers, but the *data* remains centralized, usually in the interest of protecting data integrity. However, there are situations in which the data needs to be distributed, especially when applications are deployed to regions with low-bandwidth due to spotty and/or expensive networking infrastructure. Sometimes data distribution is achieved by distributed data caching, but this often puts data integrity at risk and can be very costly and complex to manage. Lotus Domino and the Lotus Enterprise Integrator provide an ideal solution for distributing all types of data across a network because they ensure data integrity, security, and synchronization of all instances of the data set. The distribution of both unstructured and relational data using LEI provides an example of “straddling” the Lotus Domino and WebSphere lanes to meet application requirements.

4. Application life cycle

Strategic vs. Tactical

How strategic is the application? Or is it a tactical/point-solution?

This question asks you to evaluate the application purpose and life cycle. If the application is expected to have a useful life cycle of five years or more or if the application is crucial to supporting the core business, then it is strategic.

Tactical/point solutions support a relatively short-term project or business initiative. Short-term/tactical solutions may also include applications that are infrequently used. For example, an HR employee performance review application used only once a year is a tactical/point solution because it is infrequently used and is only indirectly tied to core business operations.

Due to its rapid application development environment, Lotus Domino is well suited for short-term/tactical solutions because developers can build and deploy applications in a relatively short period of time. If your organization has such applications deployed on Lotus Domino, it is not advisable to re-architect them on J2EE because it is unlikely that you would gain sufficient return to offset the development expense.

Strategic applications, on the other hand, should always be architected to deliver maximum returns over the duration of the application life cycle. So careful assessment of the application shelf-life and consideration of the factors described in this paper should help determine the appropriate lane(s) for strategic solutions.

Time to value

Does your application need to be designed/enhanced, developed, tested, and deployed within 6 months?

For example, following an internal audit, a department has three months in which to implement a change management system to show compliance with ISO 9000. There isn’t time for a lengthy development cycle, but without the solution, the department won’t reach compliance. As mentioned earlier, one of the key benefits of Lotus Domino is its rapid application development environment. It is possible to develop, test, and deploy an application in weeks as opposed to months on other application server platforms. “Mileage” may vary, of course, depending upon the complexity of the application. Many applications well-suited for Lotus Domino are relatively simple to build, especially when the developer can use one of the standard system templates that ship with Lotus Domino.

5. Environmental factors

IT enterprise infrastructure standards

Is the application subject to IT strategic policy or enterprise governance?

Many IT organizations are making strategic decisions regarding their application infrastructure to simplify systems management and to save money through server consolidation. Most companies base their strategy on the J2EE application architecture, LDAP for directory services, and relational databases for data storage. If application architecture is governed by such IT enterprise strategy, then it probably belongs in the WebSphere lane, keeping in mind that Lotus Domino may be needed for features not yet available on the J2EE platform.

Another fact to remember is that you can configure the Domino Directory to be an LDAP directory, so your existing Domino Directory will fit within IT policy, IT infrastructure, and support resources.

Are full IT resources available?

There is always overhead associated with application development, deployment, and management, including IT infrastructure (database, application server, directory, and security) and human resources for system administration, database administration, directory services, or security architecture.

For example, consider the application that needs to be deployed to a field site in a remote third world country without adequate local IT support or WAN bandwidth. In addition to infrastructure constraints, budget constraints are another factor. For example, a business unit has identified the need for a solution, but is unable to secure adequate funding for IT resources. These are just two of many cases in which there is a justifiable business need for the application in absence of available IT infrastructure and human resources.

The fully integrated (as described in the section “The Value of Domino Apps”), stand-alone application server provided by Lotus Domino would be the most suitable solution for these problems. Lotus Domino provides virtually everything your organization needs to build, run, and deploy many types of applications.

Application development skills

Do you have obtainable/available Lotus Notes and/or Java development and administration skills?

Many applications need developers to create and maintain them, and system administrators to manage them. Oftentimes, application architecture is determined by the available skills in the organization. Making an assessment of which resources are currently available or obtainable through hiring or contracting is an important factor in determining which lane choose.

The Future of Rapid Application Development

J2EE architectures offer platforms with inherent performance benefits for Web application development generally and collaboration development specifically. Many customers and application providers are already reaping these benefits in their systems and solutions. These trends foreshadow tremendous growth in the use of J2EE architectures. To facilitate this rapid growth, however, development of J2EE applications and services must develop attributes that appeal to a broader developer audience.

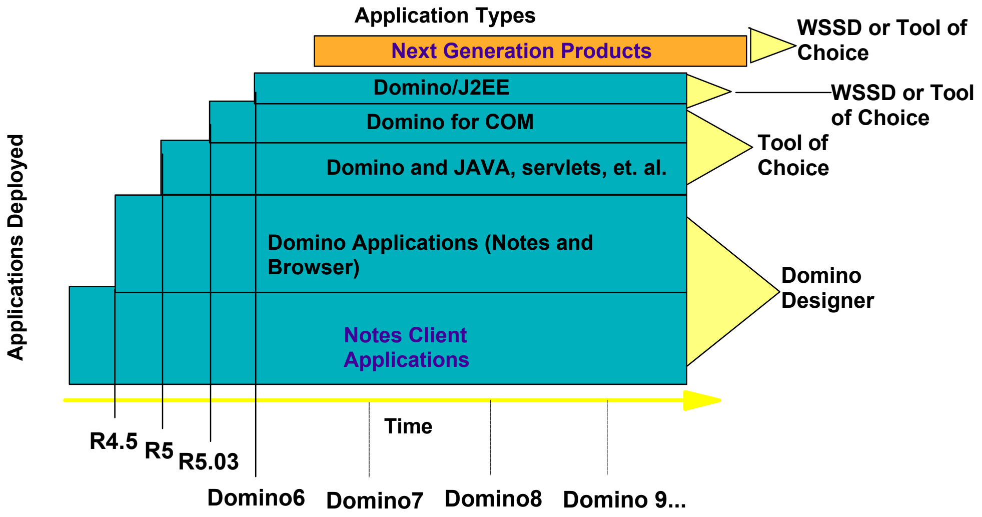
J2EE application development tools must evolve to appeal to application development communities beyond the traditional Java-savvy developers in IT departments (who primarily focus on enterprise and systems applications programming) to Web application developers in business units. To evolve requires the creation of a rapid application development (RAD) tool that empowers Web application developers to build and to connect components, data, and application business logic, while insulating them from the complexities of the J2EE architecture.

Lotus is leveraging its understanding of the needs of RAD tool users, along with the expertise gained in developing Domino Designer, to build a premiere J2EE RAD tool primarily targeted at IBM's J2EE platforms. This project is currently underway and is a cooperative development effort between the Lotus e-Workplaces and IBM WebSphere engineering teams.

The project's intended audience is the corporate Web application developer. Corporate application developers, as opposed to systems programmers or enterprise developers, generally seek to create a user experience that executes behaviors (that is, performs certain functions). The appearance and flow of the application, together with the behaviors it performs, define the application for an application developer. These developers generally start with the user interface because they think of the application first in terms of how it looks. Behaviors are then added on top of the UI, and pages are linked together to form the application

This future RAD Tool will provide an easy, visual way to develop rich Web applications that leverage the J2EE environment. It is important to note, however, that while the RAD Tool will support the easy creation of applications, it will not be limited to generating simple applications — it will benefit developers who are comfortable working in an independent manner by providing a rapid way to build the foundation of an application, then allowing complete customization through access to any and all details of the application using tools in WebSphere Studio. Lastly, this future RAD Tool will be a comprehensive RAD tool, covering all phases of application development: coding, testing, and deployment. This future RAD Tool from IBM is based upon the Eclipse open source framework and leverages the tools available in IBM's WebSphere Studio Family for many key functions, such as page design, script editing, debugging, bean creation, and so on. Lotus will continue to enhance Domino Designer as a hybrid client application development environment for Lotus Domino with utilities added over time to facilitate interaction with WebSphere tools.

The following diagram shows the supported Lotus application development models for today and for the future, with the associated application development tools for each:



*WSSD: WebSphere Studio Site Developer

Applications built for Lotus Notes currently represent the largest percentage of custom applications deployed for Lotus products. These are applications built primarily with LotusScript and the Lotus Notes formula language. The next largest deployed application model is Domino applications deployed to Web browsers or hybrid Lotus Notes/browser applications. Lotus will continue to support these application development models for the foreseeable future as long as customers are using them.

Domino Designer is the application development tool for building Lotus Notes and Domino applications. The future of Domino Designer is continued enhancement, especially focusing on features that enable the development of integrated applications and the use of the next gen collaborative services. There are no plans to port Domino Designer to WebSphere Studio or Eclipse — it will remain a dedicated RAD tool for Lotus Notes and Domino. In Lotus Notes and Domino Release 5, Lotus extended the development model by adding new APIs for Java and COM, which are important to interoperability with external applications. In Lotus Domino 6, a key feature provides the ability to build integrated Domino/J2EE applications using the Domino JavaServer Pages (JSP) tags and enhanced Java classes.

This application development model is the stepping stone to the future and provides two main customer benefits:

- Lotus Domino functionality can be integrated into J2EE applications
- Current Domino developer community will have a relatively easy way to begin working with J2EE with its existing skills set.

Lotus plans to provide a plug-in for WebSphere Studio 5 to allow developers who want to integrate the functionality of a WebSphere application with a Domino application. This plug-in gives a developer in WebSphere Studio the ability to see the design elements in a Domino database and to launch Domino Designer to modify the element as well as the ability to copy the basic structure of a form or view into a JSP using the JSP custom tags provided with Lotus Domino 6.

As described in the opening section, the future strategy introduces a new application development model that provides consistent services-oriented interfaces to all Lotus products and is based on the Java and Web services standards. The new RAD tool for J2EE will be the best way to enable Lotus collaborative services to be integrated with J2EE applications.

Conclusions: Preparing for the Future

To prepare for getting the most out of Domino as its J2EE focus grows (i.e., the next gen lane when it merges with the Lotus Domino and WebSphere lanes), consider the following:

- Build skills in JavaServer Pages, Java, and Web services technologies, especially XML. JSP and Java are essential to building high-value applications that straddle the Lotus Domino and WebSphere lanes today. Download SoapConnect from http://www.alphaworks.ibm.com and learn to incorporate Web services with existing Domino applications. In the future, these will be the foundation technologies for the next gen application interfaces, so now is the time to build skills in these areas.

- Implement LDAP for directories. If you have a Domino 5 or 6 Directory, then adopting LDAP is a matter of configuration. Lotus software will increasingly use LDAP for directory services in the future, so this will be an essential action.

- Upgrade to Lotus Domino 6 for the “stepping stone” features. Lotus Domino 6 has a number of features that enable excellent application integration with WebSphere. Lotus Domino 6 provides a new JSP tag library as an easy-to-use interface to the Domino Java objects. The Domino Java API has been enhanced for “remoting” to the WebSphere application server environment. Lotus Domino 6 also has enhanced LDAP support, native XML Java classes, optional integration with IBM Tivoli® for systems management, and tighter RDB integration through LEI.

- Reorganize development teams so that Domino and WebSphere developers work together to build integrated applications. Domino developers bring unique skills and experience in building user-centric collaborative applications. WebSphere developers have experience in systems programming that can be valuable to extending Domino applications to larger scale deployments beyond the enterprise. Through organizational alignment, these teams can tap each other’s strengths to integrate collaboration to more areas of the business and beyond the corporate boundaries. This also applies to any customer using a J2EE container other than WebSphere.

- In project planning, remember that all kinds of future solutions can have collaboration integrated for maximizing user productivity in more areas of your business than is feasible today.

IBM continues to release additional documentation, like white papers and redbooks, to keep updated on new technological enhancements. See the following redbooks for further reading related to application development and integration between Lotus Domino and WebSphere Application Server.

SG24-5955-01: *Domino and WebSphere Together Second Edition*
SG24-5331-01: *Lotus Domino Release 5.0: A Developer’s Handbook*

Redbooks are available from http://www.redbooks.ibm.com