Lotus Discovery Server 2.0
Deployment, Planning, and Integration

Understanding the Discovery Server architecture and key capabilities

Integrating Discovery Server into your environment

Planning and deployment tips

William Tworek
Anna Claverol
Philip Monson
Martin Rueckert
Steven Weissberg
George Westwood

ibm.com/redbooks
# Contents

**Notices** ................................................................. ix
**Trademarks** ......................................................... x

**Preface** ................................................................. xi
The team that wrote this redbook ........................................ xi
Become a published author ............................................. xiii
Comments welcome ...................................................... xiii

**Chapter 1. Introduction to Lotus Discovery Server 2.0** ............. 1
1.1 Essential terminology and definitions ............................. 2
1.2 An introduction to Knowledge Management ....................... 3
  1.2.1 People, places, and things ................................... 4
  1.2.2 Knowledge Management technologies ........................ 4
1.3 What the Lotus Discovery Server does ............................. 5
  1.3.1 How Discovery Server benefits your organization ........... 6
1.4 A first look at the Lotus Discovery Server ....................... 7
  1.4.1 System overview diagram ...................................... 7
  1.4.2 A guided question and answer tour .......................... 8

**Chapter 2. The Discovery Server architecture and key components** . 13
2.1 The external view of the Lotus Discovery Server .................. 14
2.2 Backend Discovery Server components overview .................. 16
2.3 The Discovery Server scheduler .................................... 17
  2.3.1 Scheduler handling of spider events ......................... 19
2.4 Discovery Server queues ............................................ 19
2.5 Spiders .................................................................. 20
  2.5.1 Where is spidered information stored ....................... 22
  2.5.2 Web spider ....................................................... 23
  2.5.3 File system spider ............................................. 27
  2.5.4 Notes spider .................................................. 29
  2.5.5 Domino.Doc spider .......................................... 30
  2.5.6 Quickplace spider ............................................ 31
  2.5.7 Notes e-mail spider .......................................... 31
  2.5.8 Microsoft Exchange spiders (mail and public folder) ..... 32
  2.5.9 Profile source spider ....................................... 33
  2.5.10 XML spider .................................................. 34
  2.5.11 Repository security, and spider ACL collection ........... 35
2.6 Metrics .................................................................. 37
  2.6.1 Metrics processing ........................................... 38
4.1.7 Firewall considerations .............................................. 84
4.1.8 Network traffic considerations ...................................... 87
4.1.9 A sample distributed Discovery Server architecture ............ 88
4.2 Discovery Server data and network security .......................... 90
4.3 Server hardware considerations .................................... 93
4.4 OS performance tuning considerations ............................... 95
  4.4.1 Tuning Windows ....................................................... 95
  4.4.2 Optimizing virtual memory. ....................................... 96
  4.4.3 Tuning Windows ....................................................... 98
  4.4.4 Monitoring performance ......................................... 100
4.5 Discovery Server-specific tuning considerations .................... 105

Chapter 5. Installation tips and troubleshooting .......................... 107
5.1 Pre-installation considerations ...................................... 108
  5.1.1 Prerequisites quick checklist .................................... 108
5.2 Upgrading versus a new install ..................................... 109
  5.2.1 Upgrading from Discovery Server 1.x ........................... 110
  5.2.2 Upgrading from a beta build .................................... 115
5.3 Installing Discovery Server 2.0 for the first time ................... 116
  5.3.1 Specifying the user account for installation .................... 117
  5.3.2 Setting up Discovery Server 2.0 ................................. 121
  5.3.3 Setting up K-map replication on the primary server .......... 126
  5.3.4 Secondary servers ................................................ 128
  5.3.5 Setting K-map replication on the secondary servers .......... 128
5.4 How to uninstall Discovery Server .................................. 130
  5.4.1 What to do if unInstallShield fails ............................. 131
5.5 Top configuration problems .......................................... 132
  5.5.1 Common name = computer name = host name .................. 132
  5.5.2 People Awareness component (Sametime) not working .......... 137
  5.5.3 Cannot access the K-map editor .................................. 140
  5.5.4 Replication problems and conflicts ................................ 141
  5.5.5 Issues with profiles .............................................. 144
  5.5.6 Problems reported in the Affinity Processing log .............. 144
  5.5.7 Spidering file systems ........................................... 145
  5.5.8 Browser-related issues ......................................... 147
  5.5.9 KD Administrators group is missing important entries ........ 149
  5.5.10 Cannot access the K-map .................................... 151
5.6 Common configuration problems related to Domino security ...... 152
  5.6.1 Access to the Domino Directory ................................ 152
  5.6.2 Server security settings are incorrectly set .................. 153
  5.6.3 Cross-certification problems ................................... 154
5.7 Additional information .............................................. 156
Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area.

Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:
IBM Director of Licensing, IBM Corporation, North Castle Drive Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:
This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM’s application programming interfaces.
Trademarks

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

Redbooks(logo)™
Balance®
DB2®
Everyplace™
IBM®
Perform™
Redbooks™
SP™
Tivoli®

WebSphere®
Lotus®
Lotusphere®
Lotus Notes®
Notes®
Sametime®
Word Pro®

K-station™
Lotus Discovery Server™
Lotus Knowledge Discovery System™
Lotus Sametime™
QuickPlace™

The following terms are trademarks of International Business Machines Corporation and Lotus Development Corporation in the United States, other countries, or both:

Lotus®
1-2-3®
Lotusphere®
Lotus Notes®

Sametime®
Word Pro®
Domino™
K-station™
Lotus Discovery Server™

The following terms are trademarks of other companies:

ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks of Intel Corporation in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, and service names may be trademarks or service marks of others.
Preface

The Lotus Discovery Server is a knowledge server. It provides search and expertise location solutions designed to ensure that the relevant knowledge and collective experience of an organization is readily available to help individuals and teams solve everyday business problems.

In this IBM Redbook we provide an in-depth discussion of the architecture and capabilities of the newest release of the Discovery Server - Discovery Server 2.0. We discuss the key aspects involved in planning for the deployment of this product, from both organizational and technical points of view.

We also discuss:

- The key elements of a solid infrastructure design to ensure scalability and reliability
- Development and maintenance of an enterprise taxonomy
- Integration of Discovery Server into your existing environment and applications
- Use of the Discovery Server API toolkit
- Ongoing Discovery Server maintenance

This book is appropriate for technologists and project managers who are involved with Lotus Discovery Server-related projects.

The team that wrote this redbook

This redbook was produced by a team of specialists from around the world working at the International Technical Support Organization, Cambridge Center.

William Tworek is a Project Leader with the International Technical Support Organization, working in the Cambridge, Massachusetts center. He manages projects which produce redbooks on various topics involving IBM and Lotus Software technologies. Before joining the ITSO, he was an IT Architect for IBM Software Services for Lotus. Prior to that, Bill was a consulting IT Architect for Andersen Consulting/Accenture. His areas of expertise include system integration and systems infrastructure design.

Anna Claverol is a senior Quality Engineer for Lotus software in Dublin. Since joining Lotus in 1998, she has been involved in many European language
localization testing initiatives, including Discovery Server 2.0, Notes 4.5x and 4.6x, Domino Workflow 2.0 and 2.1, Knowledge Discovery Server 1.0a, Discovery Server 1.1, Sametime 3.0.

Philip Monson is a member of the Lotus Discovery Server 2.0 internal Lotus Software early deployment team, serving as a content expert (taxonomy editor) for the Worldwide Sales organization. He currently works as the Technical Manager for the Lotus Executive Briefing Center, and as a Knowledge Architect on the Lotus Knowledge Architecture and Communications (KNAC) team. Phil has been with Lotus for 12 years, joining the company when the early versions of Notes were rolled out for internal use. He served as the first Notes Operations Manager in Lotus IS and then as an application developer and consultant, sharing his experiences in technical publications and on stage at Lotusphere.

Martin Rueckert is a Lotus Software Services consultant with the ISSL-Central/Germany region. He has worked for Lotus Software for over five years, starting while he was still a student studying Computer Science. Martin has a degree in Information Technology; his thesis is on “Portal Technology and KM.” He has focused on IBM/Lotus KM products, particularly WPS/K-Station and the Lotus Discovery Server. He has worked with Discovery Server since test build 3, supporting customers in the Discovery Server beta program, as well as participating on various Discovery Server pre-sales POCs. Martin has been a speaker at many Lotus/IBM KM sessions, and he is a co-author of the Discovery Server 1.0 redbook.

Steven Weissberg is a Senior Partner at Watermelon Sky, Inc., with skills in architecture, topology, security, and implementation planning. He has worked with Lotus Notes and related collaborative technologies for over a decade, and has participated in the deployment of very-large-scale Notes/Domino rollouts. In addition, he has directed the deployment of major e-commerce internet sites, and represented a major corporate client in early design discussions with Lotus about Discovery Server.

George Westwood is an IBM Software Services architect with IBM Australia. He has worked for IBM for over 14 years in a variety of product management and consulting roles, specializing in Lotus technologies. He has a degree in Computer Science. His areas of expertise include the deployment of Lotus KM products, particularly Domino.Doc, K-Station, and the Lotus Discovery Server, George has worked with Discovery Server since test build 3, and has been a speaker at many Lotus/IBM KM-sessions.
In addition to the authors, a number of people contributed significant amounts of support and guidance throughout the writing of this book. We would like to thank the following people for their contributions to this project (in no particular order):

Patrick Friel, Julia Nizhegorodova, James Goodwin, Michael Cox, Wendi Pohs, Jennie Dymacek, Steve Kelley, Dane Johnson, Lauren Wendel, David Kajmo, Scott Eliot, Tom Lowell, Craig Wolpert, Hiroyuki Miyamoto, Han H. Li, George Sprott, Chris Samoiloff, Monika Schoelzl, Dave Urban, Dave Wetherell, Cindy Regnante, Alison Chandler.

We would also like to thank the authors of the previous Discovery Server redbook, *Inside Lotus Discovery Server* (SG24-6252). That book provided a strong foundation upon which this book was built; it was written by David Morrison, Peter Northam, Martin Rueckert, and Lasisi Tabei.

**Become a published author**

Join us for a two- to six-week residency program! Help write an IBM Redbook dealing with specific products or solutions, while getting hands-on experience with leading-edge technologies. You'll team with IBM technical professionals, Business Partners and/or customers.

Your efforts will help increase product acceptance and customer satisfaction. As a bonus, you'll develop a network of contacts in IBM development labs, and increase your productivity and marketability.

Find out more about the residency program, browse the residency index, and apply online at:

[ibm.com/redbooks/residencies.html](http://ibm.com/redbooks/residencies.html)

**Comments welcome**

Your comments are important to us!

We want our Redbooks to be as helpful as possible. Send us your comments about this or other Redbooks in one of the following ways:

- Use the online Contact us review redbook form found at:
  [ibm.com/redbooks](http://ibm.com/redbooks)

- Send your comments in an Internet note to:
  redbook@us.ibm.com

- Mail your comments to the address on Page ii.
Chapter 1. Introduction to Lotus Discovery Server 2.0

This chapter provides a general overview of the key concepts behind the Lotus Discovery Server technology, including a discussion of the IBM/Lotus approach to Knowledge Management and how the Lotus Discovery Server fits into this approach.

It is geared towards the reader who is new to the Discovery Server product and its capabilities.
1.1 Essential terminology and definitions

The following terminology and definitions will be helpful to the reader who is new to the Knowledge Management area and the Discovery Server product.

**Affinities and affinity values**
An affinity is a discovered relationship between a person and a topic or a taxonomy category area. Affinity values represent the strength of the relationship between a person and a topic area.

**Application programming interface (API)**
The Discovery Server API Toolkit is a published set of entry points, methods, and classes enabling developers to interact programmatically with Discovery Server services, including K-map building, metrics processing, content search, and administrative processes. The result enables customization of search interfaces to Discovery Server-maintained data, manipulation of information or information values, and application of alternative programs or services (such as taxonomy import or alternate classification) within Discovery Server application processing services. The Discovery Server 2.0 API Toolkit provides new capabilities and APIs supporting taxonomy import and rules-based “simple” K-map content classification.

**Digital bread crumbs**
Every organization builds its knowledge every day, and not always through structured methods. “Digital bread crumbs” are the pockets of knowledge and expertise that exist in an organization or department or with an individual.

**Metadata**
Metadata includes all the details about your data, separate from the data itself. For example, metadata includes the author name, date modified, creation date, and more.

**Metrics**
Metrics represent data and analyses processes and are aimed at increasing the accuracy when evaluating organizational content and expertise to topic values. They represent the relationship between the item (document, person, or place) and associated usage information. The Discovery Server automatically and constantly analyzes collected metrics data, calculating relationships, document values, and expertise strength. As a result, the Discovery Server can deliver the information you need, rather than bombarding you with irrelevant results, as traditional searches often do.

**Spiders**
Spiders are optimized content access and extraction technologies that run on the Lotus Discovery Server. Their function is to connect to the source content.
repositories and information sources you specify, extract information/knowledge, and bring it to the Discovery Server for analyses and organization to taxonomy.

**Taxonomy**

Taxonomy is a generic term used to define a set of ordered groups or categories. It is a classification scheme, or a way to organize and present information. An example of a taxonomy can be found at most major Internet search sites, where information has been categorized into a few high-level categories.

Each top-level category contains subcategories, and each subcategory can be further divided into other subcategories, and so on. For example, if a user selected Finance as a top-level category, he might be presented with a list of subcategories related to finance from which to further select, or refine, his search.

The Discovery Server taxonomy, called the *K-map*, is automatically created by the Discovery Server. It includes comprehensive categorizations of information across source repositories and associates people expertise to categories.

### 1.2 An introduction to Knowledge Management

Before delving into the specific capabilities of the Lotus Discovery Server, it's important to first define *Knowledge Management*, a widely used term with many meanings. Lotus and IBM define Knowledge Management as a discipline to systematically leverage information and expertise to improve organizational responsiveness, innovation, competency, and efficiency. These important capabilities describe how any organization can remain competitive in an ever-changing marketplace:

- **Responsiveness** is the ability to rapidly respond to market changes.
- **Innovation** is the successful fostering of an organization’s creativity.
- **Competency** is the ability to catalog the knowledge and expertise held by current employees and make it available to others, especially new employees.
- **Efficiency** concerns the capacity to “know what you know” in order to minimize the effort wasted in reinventing the wheel.

Achieving these results is the goal of any Knowledge Management solution. The Lotus Discovery Server plays a key role in this effort by evaluating, organizing, and locating the relevant information and associated expertise that enable organizations to excel in each of these four areas.
1.2.1 People, places, and things

It's important to remember that knowledge means more than just documents or information. Knowledge is the collective heart, brain, and soul of an organization. Knowledge is derived from experience, so people play an important role in any successful Knowledge Management solution.

The overall Lotus/IBM Knowledge Management strategy states that value is created when people, places, and things are brought together in a meaningful business context. These three elements are at the core of Knowledge Management, and the Lotus Discovery Server is focused on providing an approach for searching, finding, accessing, evaluating, and using the knowledge embodied in all three of these elements.

- **People** represents the employees, customers, partners, experts, and other individuals who are central to your organization's business success. Knowledge Management products, including the Discovery Server and WebSphere Portal Family with Lotus K-station technology, provide the context and tools for effective interaction among groups of people. These technologies also offer features that identify people in terms of what teams they belong to, what they know, their degree of proficiency, and their current online status.

- **Places**, or Community Workspaces, are the real or virtual workspaces in which people come together to brainstorm, learn, and interact, such as project team headquarters, a Domino discussion or teamroom application, a WebSphere Portal “Place,” and other community workplaces within enterprise applications made accessible to the Discovery Server search services using Discovery Server APIs. The Lotus Discovery Server locates and classifies these virtual places and enables others to benefit from the knowledge found there.

- **Things** includes the data, information, and processes that are created, captured, classified, and shared across your organization, including documents; office productivity files; project, team, and research applications, ERP, CRM, or database information; Web pages, presentations, and more. Knowledge Management enables users to easily access context-specific information—regardless of its source location—and apply this key content to strategic business goals.

1.2.2 Knowledge Management technologies

Lotus and IBM have identified five essential technology categories that enable Knowledge Management. The Lotus Discovery Server provides key capabilities in several of these categories, and can be combined with other Lotus/IBM
technologies, such as the WebSphere Portal family, to provide even more capabilities. The technology categories are:

- **Business Intelligence**, as exemplified by data and text mining, OLAP, and data warehousing
- **Collaboration**, embodied by groupware products and also including synchronous and asynchronous messaging technologies
- **Knowledge Transfer**, including computer-based training, distributed learning (or e-learning), as well as live classes, seminars, and discussions
- **Knowledge Discovery**, including search tools, content classification tools, data navigation capabilities, and document management
- **Expertise Location**, encompassing expert networks, visualization, affinity identification, and other tools that connect people

### 1.3 What the Lotus Discovery Server does

The Lotus Discovery Server is a back-end server for managing your organization's knowledge. The Discovery Server provides sophisticated tools that categorize documents and user information into browsable and searchable form.

This technology finds and structures the relevant content and expertise required to achieve specific business goals by dynamically analyzing relationships among people, activities, structured content, and unstructured information. This type of information and knowledge occurs naturally within any organization every day, but is difficult to track or access.

Think of these pockets of knowledge as “digital bread crumbs.” They are the natural by-product of your current daily work routine, including the documents, e-mail, and presentations that are created by your employees. No other solution is as adept at gathering and making sense of your organization’s information and activity as the Lotus Discovery Server. It gathers and organizes the knowledge in an organization, adds contextual information to enrich the experience, and enhances the meaning of “digital bread crumbs” that already exist.

The Discovery Server also makes it easier than ever for end users to find everything about the resources they need across various systems and information sources within an enterprise.

Through a combination of automatic processes and administrative tools, the Lotus Discovery Server can perform the following tasks:

- Create a unique Knowledge Map, or K-map, of your organization's content, experts, and community workspaces
- Generate affinities representing expertise to business topical areas
- Mine skills to create complete expertise profiles
- Dynamically calculate and assign document and affinity values
- Group and organize documents to business category areas
- Search for documents, people, and topics across disparate sources

1.3.1 How Discovery Server benefits your organization

The Lotus Discovery Server provides functionality far beyond other Knowledge Management solutions on the market. It creates a new category of knowledge server, one that provides an extensive, powerful set of capabilities that translate into strong benefits to organizations and the individuals within them.

The Discovery Server 2.0 release carries forward the goals of this technology by introducing performance and scalability improvements, extensibility through its APIs, and a framework to support enterprise inter-operability requirements through the use of new APIs and the new Discovery Server XML DTD and XML Spider.

An organization can reap many benefits because the Lotus Discovery Server:
- Improves learning in an organization
- Fosters creativity and innovation
- Increases collaboration, knowledge sharing, and reuse of past work
- Makes it easy for users to search or browse the collective experience and knowledge within their organization, and beyond, then apply what they find
- Provides easy access to people and their expertise
- Connects people with the right information at the right time, enabling them to take action, make decisions faster than ever before, and deliver products and services more quickly
- Shows the relationships between information and expertise, adding much-needed context
- Supports and enhances the capabilities of key e-business initiatives and applications, extending Knowledge Management to external partners and customers
- Allows new employees to get up to speed quickly because they can access and find the resources they need
- Enables organizations to identify knowledge gaps and areas for improvement
- Integrates Knowledge Management into the core of your organization
Uses automation to streamline Knowledge Management by significantly reducing time-consuming manual categorization of large amounts of content

Ensures easy use and implementation

Respects privacy and user control of their data and enforces enterprise security policies

Allows you to aggregate content via integration with existing systems and the ability to work across platforms and systems

Learns and grows with your ever-evolving organization

The Lotus Discovery Server moves beyond point solutions to provide a comprehensive, next-generation solution for locating and organizing the relevant content and expertise required to address specific business problems and projects. It includes innovative technology that leverages the powerful combination and broad scope of Lotus and IBM. And it provides capabilities that exceed other solutions on the market, putting the Lotus Discovery Server in a class by itself.

1.4 A first look at the Lotus Discovery Server

This section is targeted to first-time users of the Lotus Discovery Server who need to get up to speed quickly on the product. Readers who are already familiar with the product may wish to skip this overview.

1.4.1 System overview diagram

Figure 1-1 on page 8 shows a typical Lotus Discovery Server system. At the top right is the Discovery Server with its K-map, people, and index databases. This system can create or add to these databases by searching through the data source examples shown on the left side of the figure. As data is added to the system, it can be managed by the K-map editors or searched by end users.
1.4.2 A guided question and answer tour

Now that you are familiar with the “essential” terminology used in describing the Lotus Knowledge Discovery System, the following question and answer tour of the Lotus Discovery Server is presented to give you a basic introduction to how the product works from a technical perspective. The objective is to put the inner workings of the Lotus Discovery Server into context, and to provide you with a foundation for the following chapters, which describe the Lotus Discovery Server components in more detail.
The Discovery Server provides services (the spiders, K-map building service, K-map Indexing service, and Metrics services) which access, manage, and analyze information from a variety of corporate and external sources. The Discovery Server’s user interface provides search and browse access to information from these sources.

**What is the Lotus Discovery Server and how does it work**

The Discovery Server is a back-end server that spiders documents and your organization’s directory to create a K-map (or taxonomy) of documents, and identifies expertise areas of profiled users and places that the end user can browse and search.

The Discovery Server has automated tools to help with the creation of the K-map. Initially, a first draft of the K-map is created by spidering a representative selection of documents or databases; this is often referred to as “selecting the initial training set.” Taxonomists (or K-map editors) then edit the initial K-map using a tool called the K-map Editor to edit the machine-generated titles and document structure (clusters) to build a working taxonomy.

Another component of the Discovery Server is metrics. This is a set of computational tasks that collect usage information and calculate document and affinity to content (expertise) ratings. First, it calculates a value for the document based on the words contained in it, and combines that with the value of the document to others as tracked through metrics. Metrics then calculates an affinity between a person and documents by monitoring user interactions to content.

When that activity calculates a strong affiliation with a K-map category area, the Discovery Server generates an e-mail to the profiled individual, suggesting this affinity to K-map content. Users get the opportunity to approve or reject proposed affinities before they are published in the K-map. The published affinities help produce the expertise affinities that are published in a user’s profile.

**What is a user profile and how are affinities discovered**

When you first set up a Discovery Server, data for user profiles is initially extracted from an authoritative people source (for example, Domino Directory from a Domino server or an LDAP server), and then loaded into a separate profile database (people.nsf) on the Discovery Server. Users can query this database directly to locate experts by skill, experience, project, education, and job type.

As documents are processed by the Discovery Server, the metrics system collects usage information and calculates affinities between people and the documents they used and authored. The metrics system continues to automatically monitor these interactions and maintain affinities to category areas as new documents are processed and affinities are approved for publishing in a
user's profile. This has the benefit of automatically tracking areas of expertise as they change—perhaps declining in one area and strengthening in others. The metrics service also keeps track of what users do (read, doclink, forward, and so forth) with the documents.

**Note**: Affinities are proposed based on relationships between people and categories in your K-map/taxonomy, and not just on a set of keywords.

**How are metrics collected from user activity**

Initially, the Discovery Server extracts the raw usage data, or metrics, from spidered data sources to create document values and to create affinities. The Discovery Server continues to track user activity on documents categorized in the K-map. Document values represent the calculated sum of all user activity on the document (such as citations, forwarding, response documents, reading, and so forth) and indicate a document's general value to the organization. In the K-map interface, documents are listed by this value metric.

Affinities are calculated by the Discovery Server metrics processes, which collect and calculate data about user interactions to all documents clustered within a K-map category. If the user has authored, read, edited, responded to, or created links to a number of documents in a category, then the metrics component calculates and proposes an affinity to the user.

Administrators can set a system-wide threshold (for example, calculate a 60% strength of affinity to a K-map category before proposing it to the profiled user) which controls the strength of the affinity based on a user's level of interaction in a category relative to all other users being tracked. A unique feature of the Discovery Server is that affinities are dynamic; they decay with time and inactivity.

**Do users have control over publishing of affinities**

The answer is yes. The design of the product emphasizes the control of the end user in making any information available to others in their profile. For example, when the individual's interaction with documents in K-map categories reaches a designated threshold, the Discovery Server sends an e-mail notification to the end user with that proposed affinity information. The e-mail notifies the end user of the affinity determined (proposed) by the Discovery Server and requests confirmation and approval or disapproval to publish the identified affinity.

If approved, the affinity appears in the user's profile document, accessible to others through K-map search. In that way, the end user has complete control over what information indicating their expertise is published to their profile and made available to others searching for subject expertise.
Once an affinity between a person and a category has been accepted, that person is then added into the appropriate category in the K-map. People with a category affinity are ranked by the strength, and this can be easily viewed via K-map interfaces like K-station and the K-map editor.

**Note:** While the default configuration of Discovery Server is to ask users to approve affinities, it can be configured to auto-publish without asking them.

**Where does full-text search fit in (or can I search the K-map)**

The Discovery Server includes a K-map Indexing (sometimes referred to as a full-text search) module which automatically indexes spidered repositories and produces a K-map Index. The K-map Index uses many of the features found in another Lotus technology called Domain Search, and is useful for users who know their subject very well and can enter precise search requests to retrieve it. A user can search for people who know about a particular topic, documents authored by a specific person, categories about a given topic, and so on, in addition to finding documents that contain the query term.

**How does the Discovery Server provide access to the K-map**

User access to the K-map information is authenticated using HTTP authentication. Furthermore, ACL information is retained, ensuring that K-map users only have access to documents they are authorized to review from their source location.

There are two methods of accessing the K-map: either as an editor, or as a user. For K-map editors using the K-map Builder and the K-map Editor, taxonomists (or human indexers) can categorize large numbers of documents relatively quickly to produce, re-structure, and maintain the K-map. One of the most innovative features of the Discovery Server is the ability of the K-map Builder service to “learn” from what humans do; the next time the K-map Builder Service processes documents, categorization is performed using the new rules of the K-map.

For a user, the K-map is accessed via a Web browser-based interface.

**Does the Discovery Server use any Domino capabilities**

The answer is yes. Domino provides rich capabilities that are leveraged in the Discovery Server. The Discovery Server basically sits on top of Domino technology.

Additionally, Domino environments typically have the richest content and the most consistent metadata, like titles, author attribution, and so forth, which makes the job of the Discovery Server much easier.
How does Discovery Server ensure data security and users' privacy

The general principle applied by the Discovery Server is that the user is always in control of the representation of himself and his information. Organizations implement Discovery Server affinity notification policies in accordance with their data privacy policies. The Discovery Server does not publish affinities or make restricted information known to the world unless the user gives explicit permission to do that.

Specifically, end users can edit their profiles, approve or deny affinity terms for publication, and control whether their e-mail is evaluated by the Discovery Server to assess the relationship of e-mail content for affinities to the organizations.

The Discovery Server also maintains all original data source security by collecting and maintaining all data Access Control Lists (ACLs). For more details on Discovery Server's usage of data ACLs, see Section 2.5.11, “Repository security, and spider ACL collection” on page 35.
The Discovery Server architecture and key components

This chapter provides an architectural overview of the key components that make up the Lotus Discovery Server: what they are, what they do, and how they interact. It first focuses on an external view of the system, including the key user interface components, and then describes the internal back-end processes that are the key elements of the Discovery Server.

Additionally, this chapter explains how Discovery Server respects data security on the different backend data repositories, and what steps need to be taken to enable Discovery Server to access backend data repositories.

The chapter is appropriate reading for all technologists who will be deploying or developing solutions based on Discovery Server, in addition to others who may simply desire a deeper understanding of the technology.
2.1 The external view of the Lotus Discovery Server

From a high-level/external point of view, the Discovery Server system is a standard three-tier architecture in which the data, business logic, and presentation/UI are technically separate components. From this viewpoint, the Discovery Server system consists of the following key components and interfaces, as depicted in Figure 2-1 on page 16.

Discovery Server Data Repositories
This component includes all the data repositories utilized by the Discovery Server system; the Discovery Server DB2 tables running on DB2, Discovery Server NSF files stored in the Lotus Domino core, and the Discovery Server XML files that are located within the file system.

Discovery Server “Backend” Services
This is the heart of the Discovery Server. It contains all of the various native (that is, C++) Discovery Server processes, running on a Lotus Domino core, and communicating via a CORBA backbone. This includes such sub-components as the Discovery Server scheduler, spiders, and worker tasks (for example, the automatic taxonomy generator).

This component is discussed in detail in 2.2, “Backend Discovery Server components overview” on page 16.

Discovery Server “Interface” Services
This component includes several toolkits for interfacing with the Discovery Server backend, specifically:

- Discovery Server Java API toolkit (Discovery Server API)
  The Discovery Server Java API is discussed in more detail in 8.4.1, “The Lotus Discovery Server API Toolkit” on page 231. Most custom-built applications in the Discovery Server 2.0 architecture are built leveraging the interface layer via custom interface servlets written to the Discovery Server API.

- Discovery Server XML DTD
  This toolkit allows for the incorporation of proprietary content that is not supported by out-of-the-box by other Discovery Server components.

- LDS Java servlets
  These servlets are leveraged by the out-of-the-box UI sub-components.
  The key out-of-the-box UI servlets that are provided are shown in Table 2-1 on page 15.
Table 2-1  Key LDS Java servlets

<table>
<thead>
<tr>
<th>Servlet name</th>
<th>Description</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>KmapServlet</td>
<td>Implements the K-map UI.</td>
<td>java\ds.jar</td>
</tr>
<tr>
<td>KDSPProfileGet</td>
<td>Implements the profile UI.</td>
<td>java\ds.jar</td>
</tr>
<tr>
<td>KDSResourceInfo</td>
<td>Used to retrieve translated resources for the browser-based setup UI.</td>
<td>java\ds.jar</td>
</tr>
<tr>
<td>RDSNSServlet</td>
<td>Name service servlet, used by the K-map editor to make a CORBA connection to Discovery Server.</td>
<td>data\domino\servlet\RDSNSServlet.class</td>
</tr>
<tr>
<td>KDSGetDbList</td>
<td>Used by the Administrator UI to retrieve a list of databases for a given server. Used when creating a Notes database repository.</td>
<td>java\ds.jar</td>
</tr>
<tr>
<td>ldapquery</td>
<td>Used by the people awareness applet to expand group names.</td>
<td>java\lp.jar</td>
</tr>
<tr>
<td>img</td>
<td>Servlet used to improve image download performance via intelligent caching.</td>
<td>java\lp.jar</td>
</tr>
</tbody>
</table>

Discovery Server applications
This component includes the actual user interfaces and user applications which leverage the capabilities of the product. This includes the KMAP UI and KMAP Editor tool provided with the product, as well as other UIs provided via the API samples, or custom coded interfaces.

The KMAP Editor is a special Discovery Server application, in that it talks directly to the Discovery Server backend services via CORBA‘, in addition to talking through servlets in the interface layer. This extra communication is required to ensure the consistency and integrity of the KMAP, as described later in 2.7.3, “K-map searching” on page 46.

The K-map editor architecture is described in more detail in 2.10.1, “K-map editor architecture” on page 53.
2.2 Backend Discovery Server components overview

This section looks with greater detail at the heart of the Discovery Server, that is, the backend services component described previously.

At a high level, the Discovery Server backend is based on an interaction between several major components:

- Semi-intelligent robots, called spiders, work on external data repositories to mine metadata about the given external data contents.
- This metadata is then placed in work queues, where it is processed by worker tasks.
- The worker tasks process the data from the queues in order to “normalize” it into a common metadata format, and make various calculations based on the data. The resulting metrics and normalized data are then written into the KMAP data repository.
Both the workers and spiders are controlled by a manager task, called the scheduler.

All of these components read and write to and from their associated work queues, which they use to track all transactions.

Figure 2-2 shows a high level view of the various components interacting. The rest of this chapter describes these major backend components in more detail.

![Diagram showing the interaction between components]

**Figure 2-2  Backend Discovery Server component interaction**

### 2.3 The Discovery Server scheduler

The Discovery Server's scheduler task is started automatically as a Domino add-in process on the primary Discovery Server. It controls work queues and spider schedules, and manages the dynamic starting and stopping of all tasks not automatically run on every Discovery Server. (The scheduler task's control applies only to Discovery Server tasks, and not Domino Server-specific tasks like the HTTP task or the DIIOP task).

This scheduler task is identified as dssched.exe when seen running on the Domino server console. It must never be stopped or reloaded.
In Figure 2-3 you can see the following processes that demonstrate how the scheduler works:

2. Monitor Completion queue for spider and worker start, stop, or progress messages.
3. If scheduled to run, issue start processing request to individual spider- or worker-queue.
4. Spiders and workers issue progress and completed messages to completion queue. If errors occur, issue corresponding problem report to queue.
5. Repository-specific spider state messages are relayed by the scheduler to the LDS Administrator application discoveryadmin.nsf.

Note: Always use the command `tell DS shutdown` to shut down the server. Shut down secondary servers first, before shutting down the primary server.
2.3.1 Scheduler handling of spider events

Spiders are handled in a different way by the scheduler than other consumer tasks. Since spiders can affect other systems apart from the LD Server and spiders can take long periods of time to process their scheduled repositories, it is desirable to be able to stop spider processing at any time. An administrator can easily perform this task by typing `tell *SPIDER quit` on the server console (where `*SPIDER` is the name of the specific spider to stop).

Since it is very likely that a spider is being stopped in the middle of processing a repository, context information is saved to enable the spider to return to the point of processing when it is restarted at a later time. This information, along with a yield message, is placed on the completion queue.

However, if a spider is actually stopped by quitting the spider task as described earlier, other spiders processing different repositories but similar data types are also stopped. To avoid this behavior, an administrator can stop a single spider from processing in the Lotus Discovery Control Center, by clicking the box next to the desired data repository and selecting `Stop processing or remove from queue` from the `Actions` menu. Doing so will let the scheduler send a yield message on the spiders notification queue. The spider will pick this message up and stop processing the repository.

The spider task is the only process for which the scheduler has this yield capability.

2.4 Discovery Server queues

The most important control files for all services in the Discovery Server system are the `.queue` files. They control which data repositories are in what state of process, hold information in temporary storage to be passed on for further processing, and even enable load balancing between a large number of threads working the same data repositories.

Discovery Server uses the following queue files:

- ATGOutOfBandClassifyQ: Queue written by ATG. Read by ATG. Contains docs to be moved to "uncategorized"-cluster.
- CompletionQ: Queue written by any consumer task. Read by Scheduler. Contains Start/Stop messages.
- LDSATGCmapWorkQ: Queue written by all CMap "workers". Read by ATG. Contains change messages from CMap to ATG.
- FileSysSpiderQ: Queue written by the Scheduler. Read by the Spider. Contains a list of file systems to spider.
- NotesSpiderQ: Queue written by the Scheduler. Read by the Spider. Contains a list of Notes NSF repositories to spider.
- ProfileSpiderQ: Queue written by the Scheduler. Read by the Spider. Contains a list of profile repositories to spider.
- WebSpiderQ: Queue written by the Scheduler. Read by the Spider. Contains information about which seed URLs to spider, what maximum number of link levels should be traversed for each URL, user ID and password to be used for each URL, and other preferences by URL.

Every Web spider also uses its own private .queue file, made unique by concatenating the REFERENCE ID to the standard Web queue name. This private queue file stores a set of URLs already traversed to prevent the redundant traversing of them.

- XMLSpiderQ: Queue written by the Scheduler. Read by the Spider. Contains list of XML repositories to spider.
- ExchangeSpiderQ: Queue written by the Scheduler. Read by the Spider. Contains list of exchange repositories to spider.
- ATGCatToMetricsClassifyQ: Internal queue written to and read from by ATG.
- FSSpiderMgrWorkQ

All queues are located in the file system in the \DS\Data\ directory.

### 2.5 Spiders

A Discovery Server spider is another Domino Server add-in process, one which invokes multiple threads to explore different repositories. Different spider types provided with Discovery Server are designed to extract content from various content repository types. Once a spider process is started, it can start any number of threads to explore different repositories.

The types of repositories supported are identified in Table 2-2.
<table>
<thead>
<tr>
<th>Types</th>
<th>Versions</th>
<th>Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domino.Doc</td>
<td>3.1 and 3.0</td>
<td>One data repository form needs to be defined for each Domino.Doc file cabinet. This allows all binders and doc databases in the file cabinet to be spidered.</td>
</tr>
<tr>
<td>Exchange e-mail (personal e-mail folders)</td>
<td>Exchange 5.5 on Windows NT 4.0</td>
<td>One data repository form needs to be defined for each private folder only for the purposes of finding affinity evidence in support of public K-map categories.</td>
</tr>
<tr>
<td>Exchange public folder</td>
<td>Exchange 5.5 on Windows NT4.0 Server</td>
<td>One data repository form needs to be defined for each public folder; this allows potentially all sub-directories in the folder to be spidered.</td>
</tr>
<tr>
<td>Notes database</td>
<td>Domino 4.6.x and 5.x platforms</td>
<td>One data repository form needs to be defined for each database.</td>
</tr>
<tr>
<td>Notes e-mail (personal mail files)</td>
<td>Domino 5.0 or higher environments.</td>
<td>One data repository form needs to be defined for each private file only for the purposes of finding affinity evidence in support of public K-map categories.</td>
</tr>
<tr>
<td>QuickPlace</td>
<td>2.06a and 2.08</td>
<td>One data repository form needs to be defined for each QuickPlace; this allows all rooms associated with the QuickPlace to be spidered.</td>
</tr>
<tr>
<td>Windows file system</td>
<td>NTFS, FAT, FAT32, Novell NetWare using Client Services for NetWare or other third-party solutions</td>
<td>One data repository form needs to be defined for each public folder; this allows potentially all sub-directories in the folder to be spidered.</td>
</tr>
<tr>
<td>Web</td>
<td>Any repository within an intranet firewall and using an HTTP URL as its locator</td>
<td>One data repository form needs to be defined for each main Web page.</td>
</tr>
<tr>
<td>XML</td>
<td>XML must conform to the Discovery Server XML DTD</td>
<td>After receiving a path on the network drive, the XML spider processes files stored in the specified location in a manner similar to the file system spider, except only .xml files are processed. All other file types are ignored. An XML spider does not retrieve files from sub-directories.</td>
</tr>
</tbody>
</table>

The queue of repositories that are scheduled to be spidered or re-spidered for changes is populated by the Discovery Server Control Center scheduler. The scheduler constructs each request from the data repository forms in the
dashboard.nsf database. The scheduler collects the information from the Data/Repository (which is specific and different for each repository) and Data/Spider Settings section (which is general and the same for all spiders of that type).

A spider polls discoveryserver.nsf (which is named “Discovery Server Control Center” and stores metadata about the repositories to be spidered) to determine the number of concurrent spider threads, whether it is enabled and when it is scheduled to be active. When a spider is active, it checks the work queue for a repository to process.

Processing is spider-specific, and each spider has different techniques to traverse and find all the content in a repository. However, all data repository spiders read data from the data source (that is, file systems, Web sites, and Domino databases), process the data, and then write “normalized” data to the metrics DB2 tables.

Each entity (document, file, or URL) is processed via entry into the K-map (stored physically in DB2 tables) and relevant information is converted into XML and written to three queues which are processed by ATG, metrics, and full text worker processes.

The XML written to the queues contains either control or status information (meaning start and stop messages for a given repository, written to the completion queue), or information from each entity or document processed. Messages in the completion queue are processed by the scheduler that controls when a repository should be spidered next, and are also displayed in the Lotus Discovery Control Center (discoveryadmin.nsf).

### 2.5.1 Where is spidered information stored

The spiders store data repository content and organizational information from the repository as normalized XML, which is distributed to four queues, as follows:

- **TaxonomyQueue**, where taxonomy context information and document information is stored
- **FulltextQueue**, where full text context information and document information is stored
- **MetricsWorkQueue**, where server transactions, metrics context information, document information, and database transaction information is stored
- **Completion Queue**, used by the scheduler for updating repository information: start and stop information and information for re-queueing and cleanup purposes
When scheduled to run, the queues are processed by the corresponding worker tasks and the scheduler. These tasks read the information from the corresponding queue and either write the data to the appropriate databases, or initiate necessary actions on the corresponding spider tasks.

2.5.2 Web spider

The Web spider is the most complicated of the five spider types because it has to traverse a Web site by links (anchors), and not by directory structure. This makes the occurrence of previously spidered content in an unsinded link tree possible, since the internet Web sites have no defined start or end.

To avoid the redundant spidering of already traversed URLs, the Web spider maintains multiple queue files, in a fashion similar to the file system spiders’ handling of queues described later in this chapter (2.5.3, “File system spider” on page 27).

The Web spider is started and stopped by the Discovery Server scheduler and given a certain time window by the user-defined spider schedule, during which the spiders will process their private work queue. It can be stopped manually by using the standard `tell .. quit` command within the Discovery Server Domino console.

**Note:** The `tell .. quit` command will stop the spider, saving its status, but may end up stopping more spiders than you intended. See 2.3.1, “Scheduler handling of spider events” on page 19 for more details on stopping spiders.

When running, the Web spider examines its work queue. If it recognizes a “to-do” item, it removes it and performs the following tasks:

**Web spider control instance**

1. Check whether URL already exists in URL Table.
   a. If not, add URL to URL Table, all other values are set to NULL.
   b. If it exists, continue.
2. Use a unique private work queue. Check whether it already exists, and whether the spider is being restarted.
   a. If normal restart message, then queue must exist and it does not need to initialize the `<set>` which was stored in the context.
   b. If error restart message, then we have lost the `<set>` information and we just begin processing the URLs in the queue.
   c. If not being restarted, the queue should be empty.
3. Create a <set> and pass this to child threads. The <set> is used to prevent redundant traversals of URLs.

4. Process include and exclude lists and create internal regular expression data structures.

5. Create <n> child threads. (<n> is defined by the user in Discovery Control Center.)

6. Add the starting seed URL to the private work queue. It contains, in addition to the URL, the remaining levels to traverse.

7. Wait until the queue is empty or it is time to yield (outside schedule window) and the child threads are idle. Write progress information to the completion queue.

8. Write a completion message to the completion queue:
   a. If done, write a completion message.
   b. If leaving because the spider needs to yield, write a restart message which contains all the data from the <set>.


Web spider child threads
1. Check to see if the schedule window is still valid. If so, continue; if not, wait in idle state.

2. Read a task from the private queue.

3. If empty, wait and go back to step 1.

4. Set state to busy.

5. Get the URL and check whether it has changed since last processed.
   
   To check whether a URL (usually the document this URL represents) is new, the Web spider retrieves the Last-Modified field from the Web server and compares this value with the one stored in the metrics.
   
   If it is new, then it will need to be processed. If it is already in the DB2 URL table, then check the date. If the server doesn't support modified date, then fall back to size or other techniques. If it has changed, convert to XML and dispatch to Full Text, Taxonomy, Metrics, maybe 1.

6. Parse the URL for links and perform the following for each link:
   
   Determine whether this link should be added to the queue by checking robots.txt to see if the spider is allowed by the Web site administrator to spider the scheduled URL. See “Spidering BOF Report,” by Mauldin et al., for information about the robot exclusion standard. It is available at:
   
   http://www.w3.org/Search/9605-Indexing-Workshop/index.html
Cache the last robots.txt processed and, if working on the same server, don’t fetch and parse it again.

7. Check whether this URL is included in the <set>. If it is, then it has already been traversed. If not, add it to the set.

8. Check that remaining levels to traverse is not zero.

9. Add a new task to the private work Queue. In addition to the URL it contains the value for remaining levels to traverse. Remaining levels to traverse is calculated by subtracting one from the level passed in.

10. Set state to idle.

Web spider metadata considerations
When spidering Web sites, it is important to check the structure and types of HTML used on the Web site first. For example, if the content is presented using Java applets or Flash components, the Web spider will not be able to correctly read the presented documents at all. To have the same rich amount of metadata spidered, the Web spider needs the html-documents to provide certain html-elements, including so called “meta tags,” such as the “Last-Modified” data tag.

For more information on the Last-Modified date, refer to RFC2616 at:

http://www.ietf.org/rfc/rfc2616.txt?number=2616

The Web spider supports (parses) these HTML elements and normalizes them to XML:

- Meta
- A (Anchor)
- Text
- Frameset
- Title

It also supports these meta tags:

- codepage
- title
- subject
- author
- keywords
- comments
- category
- notes
- manager
- company
Therefore, you can map meta tags to Discovery Server entities to have Web-spidered contents added to the taxonomy with a rich set of meta-information, just like you can in a Domino database.

**Web spider performance considerations**

If you have problems spidering very large Web sites, you may want to change the detail level of the Web spider log file. You can change the way errors and warnings are logged by adding to the notes.ini file the following line:

```plaintext
LDS_WEBSPIDER_WARN_LEVEL=<level of logging in values 1, 2, or 3>
```

where:

- **1** = log nothing
- **2** = filter the logged messages
- **3** = log everything

**Important:** Setting LDS_WEBSPIDER_WARN_LEVEL=3 may result in very large Web spider logs.

An additional performance consideration exists around the Web spider’s usage of memory. In the current release, the Web spider does not write this list to disk, but keeps it in memory for the time of the current traversal. Therefore, depending on the size of the Web site you have defined as a data resource, this list might consume large amounts of RAM and virtual memory while the spider is running.

The current recommendation for Web spider settings is *Do not follow links to other servers* when levels to be spidered is set to a maximum of 5.

**Web spider security considerations**

The Web spider can be supplied with a user name and corresponding password to spider password-protected Web sites by entering this information on the corresponding data repository page in the Discovery Server Control Center.

**Note:** The Web spider implements the standard “Basic” (Base64) HTTP authentication for Web servers. Customized login dialogues which handle logins using form-based html pages or Java Script/Java/PHP/CGI applications are not supported.

Keep in mind that, for security reasons, this user ID and password will not be stored in the K-map. Users trying to access spidered Web sites via the K-map will still be prompted to supply their private user ID and password for access to restricted Web sites.
2.5.3 File system spider

The file system spider is a very useful tool for spidering even unstructured Windows file system repositories with large amounts of files using different file types, while still respecting the security settings this file system may provide.

The file system spider is passed a networked file resource and recursively works the directory structure on the path. Just like the Web spider, the file system spider keeps track of previously spidered files since the files could have been moved, copied, or altered. To avoid redundant re-spidering of these files, timestamps and modified dates are processed and monitored in a private file system spider list. The file spider is passed a root directory UNC and recursively crawls all directories and files it can find. While doing so, it creates a queue file for already spidered files called FSSpiderWorkQ. The “root” directory where the spider begins processing is set according to the root directory defined within the file repository record in Discovery Server - it does not necessarily have to be the actual file system root.

The FSSpiderWorkQ is a temporary queue created at the beginning of each File spider traversal. A two pass approach is used. The first pass traverses the file system and records every file needing processing in the queue by saving the file name. The second pass reads from the queue and processes each file. This model is used to make yield/restart work easier, and also to lay the groundwork for a multi-threaded file spider in a later release of Discovery Server. If the first pass is not finished because a yield request is handled, then the existing workQ is discarded and created again the next time. Once the list of files is fully created (FSSpiderWorkQ) by a successful first pass traversal, then yield requests are easily handled. The file spider simply restarts by picking up processing from the next file name in the FSSpiderWorkQ.

The FSSpiderMgrWorkQ is used to track available FSSpiderWorkQs. Each spider asks the WorkQManager for a local workQ name. When a spider is done with a queue it gives the workQ name back to the manager. If the manager doesn't have any names in the FSSpiderMgrWorkQ, then it generates a new name for a workQ.

Supported file types
The file system spider can process all file types supported for conversion by the Verity KeyView file viewer, also used in Domino R5. Among the file types supported are:

- Adobe PDF
- Microsoft Word MAC V4.x to 6.x, 98
- Microsoft Word 95, 97, 2000
- Microsoft Word PC V2.0 to 5.5
- Microsoft Word for Windows V2.x, 6.0
**File system spider performance**

When spidering file systems, keep in mind that the file system spider is very resource-consuming (especially network bandwidth) because it opens every file repeatedly for the initial spider run of the given data repository.

**File system metadata considerations**

It is not necessary to individually map certain file type metadata fields to the repositories field map, since this is already done by the file view filter. Any metadata in your original data file will automatically be passed on to the corresponding metrics consumers.

**File system security considerations**

When a file system spider runs, it collects the access control information from the file systems that it spiders. On standard Windows file systems, Microsoft provides an interface that allows the spider to read the access control information, as long as the spider has sufficient access. The user account that the file system spider runs may use Full Control for simplicity, but the minimal permissions the spider requires are:

- **At the network share level:**
  - Read
  - Change access can be set to Deny

- **At the file directory level:**
  - List Folder/Read Data
  - Read Attributes
  - Read Extended Attributes
  - Read Permissions
  - All other access can be set to Deny
To collect ACL information about NTFS file systems, the spider must have permission to read access control permissions on the system being spidered. This access can be set on the folder being spidered by its administrator. When spidering an NTFS file system, ACL information is obtained from file share ACLs, directory ACLs, and file ACLs. The Microsoft file system APIs are utilized to collect the file ACL information.

FAT and FAT32 file systems do not have directory and file level ACLs. The only access information used is the share access lists.

**Restriction:** The Discovery Server 2.0 file system spider cannot collect ACL information from the native Novell file system. The product documentation, under the file system section, recommends one workaround for this issue. An alternative could be to utilize Samba technology.

### 2.5.4 Notes spider

The Notes spider’s job is similar to those of the file system spider and the Web spider. It gets a list of new or altered documents in a Notes database and spiders those documents. If the database is a new data repository, it processes all documents during the first pass of the spider.

Timestamps are compared to check whether a document has been altered since the last Notes spider ran. The list of documents to be spidered gets passed to a conversion function which converts all items to XML. This normalized XML stream is passed to the metrics queue, K-map indexing, and taxonomy queues.

**Notes spider field mapping**

If a Notes document in a Notes database contains attachments, there are some important features of the Notes Spider to know about:

- If the attached file does not contain an author, the author of the container (that is, the Notes document) will be used.
- If the attachment does not have a title, the title of the container is combined with the attachment name.
- You do not need to manually add attachment fields to the field map. This is done automatically for all Notes databases. However, if your company stores attachments in a Notes database application and you want these attachments to be spidered, be sure the database you spider actually contains these attachments since links to other databases are not traversed by Discovery Server.
Notes spider performance considerations
If your Notes data source views contain a large number of documents that do not contain valuable text for the K-map, or you just do not want them to be in the K-map, create a hidden view in your Notes database called something like $DominoSpider, where you select the documents you want spidered. Then set the spider to retrieve this view only within the Notes data repository record within Discovery Server.

This will significantly enhance the quality of your K-map and the speed at which these data sources are spidered.

Notes spider security considerations
Notes spiders collect access control lists (ACLs) for both repository- and document-level access, also referred to as reader fields (if document access restrictions exist). When ACLs are changed in the database, the spider will pick them up on the next spider scan. The Notes spiders need Reader access to the repositories for spidering and collection of ACL information.

2.5.5 Domino.Doc spider

The Domino.Doc spider is certified for Versions 3.0 and 3.1 of Domino.Doc. It is very similar to the standard Notes spider, but extends its core features to some Domino.Doc-specific repositories:

- Domino.Doc repositories are defined at the File Cabinet level, and all binders and document databases within that File Cabinet are traversed.
- Binder documents with only one attachment are treated as one entity.
- Forum (discussion) documents are spidered.
- Drafts and bookmarks are ignored.
- The Administrator has the option to spider all versions or only the latest versions of documents.
- Documents with the archive flag set are excluded from spidering; attachments, if previously added to the K-map, are removed from the K-map.

Domino.Doc spider security considerations
In order to successfully spider a Domino.Doc application database, you need to ensure that the Discovery Server's name is included in the Domino.Doc Site Administrators group.

As the Domino.Doc cabinet is read, the ACL information found in the cabinet will be used as the K-map Repository ACL. As all of the binders are found, ACL
information for each binder will be saved and, along with the ACL information found in the Domino.Doc binder document, will be used to build the K-map Document ACL.

2.5.6 Quickplace spider

The Quickplace spider is certified for Versions 2.06a and 2.08 of Quickplace. It is very similar to the standard Notes spider, but extends its core features to some to Quickplace-specific sites:

- Quickplace repositories are defined at the main.nsf level; all rooms associated with that Quickplace are traversed.
- Embedded pages are treated as one entity.
- System documents, the member list, and the search index.nsf are ignored.
- Contains support for some Quickplace-specific field mappings

Tip: Be sure to spider Quickplaces on the Domino database level. Do not try to spider Quickplaces using the Web spider, since only the Domino database-level Quickplace spider will provide the features listed here.

Quickplace Spider security considerations

In general, the Discovery Server's security model is that if the user doesn't have access via the source, they won't see the document in the K-map either. In release 1.0, Discovery server respects database ACLs and document Reader fields. In release 2.0, at the repository level, Discovery Server honors the Quickplace ACL instead of the database ACL. A Quickplace can have several databases, but ACLs on the main database determine the repository access. Document access is determined by reader lists on the room documents, as well as the reader list on the document. There can be a hierarchy of rooms, so ACL information at all levels is collected. Users defined only in the Quickplace are not going to have access to the data through Discovery Server; only users that are defined in a known directory can gain access to Quickplace data through Discovery Server. The primary reason for this is that Discovery Server authenticates all users against a single directory (or set of directories linked by Domino directory assistance).

2.5.7 Notes e-mail spider

The Notes e-mail spider works very similarly to the standard notes spider. The only difference is that data in private e-mail databases will never appear in the K-map (since the contents of the documents are not stored in the DB2 tables). This means the documents will not be searchable via the full text index, nor browsable through the K-map user interface.
Instead of storing the data in the K-map, the Notes e-mail spider performs the following tasks on e-mail data:

1. It mines private mail files for possible affinity (category) mappings. By default all mail files are not spidered, and users must approve this mining of their mail file. However, this can be overridden at an administrative level, and all mail files can be spidered regardless of user acceptance.

2. If affinities are discovered in the private mail file that correspond to the K-map categories, they are either automatically published or the user is sent a mail message requesting the user to approve or deny the proposed affinity (depending on how you have set up affinity publishing policies in Discovery Server Control Center).

**Note:** It is not possible for users to be suggested an affinity for which there is no corresponding K-map category from other data since it is not possible for a category to be created only from e-mail content.

See 7.3, “People metrics (affinities)” on page 192 for more details on how the affinity process in Discovery Server works, and the various options for enabling it.

### 2.5.8 Microsoft Exchange spiders (mail and public folder)

Discovery Server 2.0 is shipped with a spider that is certified to mine document and affinity data from Exchange 5.5 servers.

To achieve this, the Exchange spider leverages the Messaging Application Programming Interface (MAPI). The spider is very similar to the Notes spider, although Exchange-specific issues apply here.

**Exchange spider access considerations**

For the Discovery Server Exchange spider, the Windows NT account used to run the spider must have administrative privileges on the Exchange server directory. It is recommended that you use View Only Administrator privileges.

**Exchange e-mail spider**

Generally speaking, the Exchange e-mail spider does the same job the Notes e-mail Spider does. It crawls private Exchange mailboxes and compares the possible affinities discovered there against the public K-map. If affinities are found in those private mailboxes that match categories in the K-map, the affinity is either automatically published or an e-mail is sent out to the user, asking the user to approve or deny the proposed affinities.
Exchange Public Folder spider
The Exchange Public Folder spider crawls documents in an Exchange Public Folder and adds the documents to the K-map for all allowed users to browse and search.

Exchange Public Folder spider security considerations
To be able to spider an Exchange Public Folder, in addition to the access rights an e-mail Exchange spider needs, the mailbox specified in the repository profile must be made an owner of each public folder that you want spidered. This setting is made in the Discovery Server Control Center, and is needed to get a full ACL and to ensure that attachments in the folder's messages can be read.

2.5.9 Profile source spider
The profile source spider is not a content spider that mines for data; instead, it is used to extract people information from directory resources. The repository types used for this extraction process are either Domino directories or LDAP directories.

Similar to all other spiders, it writes data drawn from a repository to an XML stream, which is then mined for information by a metrics add-in task. The profile source spider is able to spider various repositories to mine them into a single people.nsf profile representation, aggregating the information.

When spidering an LDAP directory, the profile source spider spiders the metadata in the following LDAP fields, which are also retrieved by the Lotus Discovery server from an LDAP directory server to authenticate a user:

- AltFullName
- Certificate
- Firstname
- Fullname
- HTTP_Hostname (Server)
- InternetAddress
- Lastname
- Listname
- Location
- MailAddress
- MailDomain
- MailFile (Person)
- MailServer (Person)
- Members
- PublicKey
- ShortName
- userCertificate
2.5.10 XML spider

The XML spider is another content spider, but one that is geared towards incorporating proprietary content that is not accessible via the other Discovery Server spiders. It reads XML data in, and then delivers normalized data to the consumer queues.

Generally speaking, the XML spider enables you to import any type of data that has been previously converted to on-disk XML files, as long as those XML files use the Discovery Server 2.0-defined XML DTD.

Discovery Server uses the IBM XML for C++ parser (XML4C) to manipulate the XML code, which is based on Apache's Xerces-C XML parser. The Xerces-C XML parser is a validating XML parser written in a portable subset of C++. XML4C integrates this Xerces-C parser with the IBM International Components for Unicode (ICU) and extends the number of encodings supported to over 150.

For more information on XML4C, refer to:

http://www.alphaworks.ibm.com/tech/xml4c

More details on the usage of the XML spider to integrate Discovery Server 2.0 with different types of data sources can be found in 8.3.3, “How the XML spider works” on page 220.
2.5.11 Repository security, and spider ACL collection

The spider tasks are responsible for gathering original data access control lists (ACLs) to ensure that data remains secure within the Discovery Server system. ACL data gathered by the spider will be of a format specific to the system from which the data was taken and will be opaque to the spider (for example, Notes ACLs are in Lotus Multi-Byte Character Set (LMBCS), and NT file ACLs are in UNICODE).
Two levels of access are collected:

- Access rights to the repository
- Access rights to the document

Repository access rights take into account server access restrictions. Thus, someone who is in a Notes database access list but who is not allowed access to the server will be considered to not have access to documents from the database.

This may seem ambiguous since a user may be able to access the same Notes database on one server but not on another. However, this is standard Domino security. It requires that the Discovery Server be set up by its administrator to not spider data from an overly restricted server if there is a better choice.

**Important:** Do not spider data from an overly restricted server since this could result in a K-map taxonomy with data usable by only a small subset of users.

Repository ACL data is packaged for transmission to the Content Map to be associated with repository-specific information. Document ACLs are packaged separately and included with other document meta information. An ACL package consists of header information in Unicode followed by an opaque ACL byte stream. The document and repository byte streams are interpreted together by the method appropriate to the repository at the time of access checking. Some access results are cached with the user context to reduce the time burden of access checking.

This context is created when the user logs in and is attached to an ACL access context object in Content Map server memory, which will control access checking of all Content Map documents. This object and the user context will be removed when the user's session is destroyed.

**Maintaining ACLs during data access**

Before allowing a document title to be displayed, the access rights of the viewer must be checked against the ACLs that are associated with the repository and the document. The ACL information is kept in the byte stream that was obtained by the spider. A class like the one shown in Figure 2-5 is used to do the access checking. There would be one of these for each user CMap session.
This class is created with a pointer to the Discovery Server user context (previously loaded into memory). For each document, a call to AccessAllowed is made with pointers to the repository and document packages. Headers of these packages will contain the repository name and type, so that control can be dispatched to the appropriate access-checking routine. The repository and document data will contain special flags denoting default or anonymous access, so that detailed checking can be avoided where possible.

Results of the access checking are cached by repository name so that random access of repositories does not cause redundant ACL checking. Some document access caching will be done as well. The return from AccessAllowed() would determine if a document should be exposed or not.

2.6 Metrics

The Discovery Server metrics process is a very complex, multi-threaded server task running various sub-task as “threads” of the main metrics task. Figure 2-6 on page 38 provides an overview of the various metrics tasks that are involved, and their interaction with other Discovery Server components. The remainder of this section discusses each of these metrics “sub-tasks” in more detail from an architectural viewpoint.

Specific details to help you understand the purpose of metrics, their usage, and the tuning of these capabilities can then be found in Chapter 7, “Metrics and affinities” on page 185.
2.6.1 Metrics processing

As you can see in Figure 2-6, the Metrics add-in task consists of various workers and one spider. However, an easy way to look at the metrics architecture is to consider metrics logically divided in two parts:

- Metrics processing, described in this section
- Metrics collection, described in the next section

Metrics processing consists of the following two tasks:

- AffinityProcessing, AffinityCalc task

The AffinityCalc task, which is a thread in the MetricsAddIn task, calculates the affinity a person has to a certain topic (in the form of a value describing the person's interest in a certain topic or K-map category). The calculation of
affinities is an ongoing process that continually recalculates the affinity value based on the affinity value this individual had to this certain topic before.

- This affinity value gets passed on to the MetricsProcessing task, which (if set to do so in the Discovery Server Administration interface) initiates the mail workflow with the user to let the user accept or decline the proposed affinity. If the affinity proposal is accepted by the user, this task also publishes the affinity to the people profile database (people.nsf) and stores it in the corresponding DB2 tables of the metrics database.

Note: In addition to having the ability to decline proposed affinities, users can approve or decline having their mail files spidered to determine any relationships between Domino mail content and K-map category areas to strengthen affinity calculations of profiled users. This is a necessary control for end users, in particular because some countries require that end users be notified that data is being collected electronically and must approve the use of that collected information. For more information about data privacy, consult: http://www.ibm.com/privacy

2.6.2 MetricsCollection

MetricsCollection is actually a name for the MetricsWorker (a thread of the MetricsAddIn task) that accesses the MetricsWorkQ to read a stream of XML documents from it and mine each of these XML documents for certain raw metrics data. When the XML of each document has been mined, the metrics data is written and updated in the metrics database (tables in DB2).

The raw XML documents in the stream extracted contain entities for each of the fields, defined in the mapping documents for your spidered repositories (for example, $Global map).

Document value in K-map UI

The document value is an approach to evaluate the content of a document in the K-map. This value is controlled by the following triggers, which you can re-sort in terms of their relative weights on the computed document value:

1. Links to a document
2. Links from a document
3. Responses to a document
4. Times a document has been opened using K-map
5. Recency of the last update to a document
The top-most trigger in this sorted list represents the value with the highest weight on the calculation of the document value. The higher this number is, the more useful the associated document is meant to be for the users. You can change the order of these triggers in the metrics settings.

**Document fit value in K-map editor**

The *document fit value* (which can be viewed using the K-map editor tool) is a computed number, representing the fit of a particular document into a certain category obtained though the K-map building clustering process. This value is actually the cosine distance of a document vector in the K-map document space and the very center of a K-map cluster (the so called centroid), which is, in fact, a category in the K-map.

![Image of Document Cluster in 3D Vector Space](image)

*Figure 2-7  Document cluster in 3D vector space*

To compute this cosine distance (fit value) between the document vector and the center vector of this particular category, the angle between the two vectors is computed and the cosine of this angle, scaled by one hundred, is applied to that value. Zero means this category has not been retrained, one indicates a bad fit value, and one hundred represents a perfect fit of this particular document in that category.
### 2.6.3 Profile synchronization

The main purposes of the Profile Synchronizer are to create new profiles and track updates to the organizational and contact information for existing profiles.

The first part of profile synchronization is mainly processed by the metrics affinity tasks (metrics AffinityCalc task and AffinityProcessing).

Data in the Metrics database is processed by the Metrics AffinityCalc task, calculating the affinity values between people and categories. Discovered affinities are then proposed to AffinityProcessing, which sends out an e-mail with the proposal to the corresponding users. If a person accepts an affinity proposal, that affinity is published by AffinityProcessing to the people directory database.

The second part, updates or creation of people profile documents, is done by the ProfileMaintenance task, which reads update requests from the Directory Sync Work Queue and publishes or updates people documents in the people.nsf repository.

Affinities are, as already stated, calculated and published by the Metrics AffinityCalc task and the AffinityProcessing task.

To get a published affinity to a certain topic, a person has to demonstrate a strength of relationship to existing K-map category areas via their interactions with content:
If a person “acts” on documents more often than other users, the metrics affinity calc task proposes an affinity between the person and the categories where the documents were found that triggered the affinity calculation task to discover an affinity.

So, translated into a real-world use case, if a person is discovered to have an affinity threshold of 85, this person has worked more on documents of a certain category than 85 % of all Discovery Server users. At that point, an e-mail is sent to the profiled user asking the user to approve or disapprove that category term being published to their profile as an affinity.

2.7 K-map

The following sections explain how the K-map's automatic taxonomy building services (also referred to as the Automatic Taxonomy Generator, or ATG) and its automatic document clustering technology actually work.

2.7.1 K-map building

In order to build the initial Knowledge Map for the Lotus Discovery Server, the Discovery Server Scan task continually scans the LDSTaxonomyWorkQ.queue file for new XML content being written to that queue by one of the three Discovery Server spider types.

Data normalization via InXight tools

The spiders, as part of the processing and normalization of the data mined from the data repositories, call multiple tasks from the InXight tools SDK. These InXight tasks change the data's word stream to a format ATG and other
consumers can work on. The Inxight tools that are called for this purpose by the InXight Wrapper perform the following normalization tasks:

1. The spider compares the documents against all stopwords lists on the Discovery Server, words found in a document and in the stopwords files are removed from the stream. All rich content, such as formatting, is removed as well.

2. The InXight tool **tokenizer** marks the remaining words as tokens, and spaces between words (tokens) are removed. It also removes the following characters and words:
   - Numerals
   - Punctuation (such as "~!@$%^&*()+-=\["]\;":",.,<>/?", )
   - Possessives
   - Addresses (e-mail, URL, network addresses, file locations)
   - Time phrases such as *Monday* (Only applies to Chinese language)
   - Conjunctions such as *and* and *or* (Only applies to Chinese language)

3. The InXight tool **stemmer** reduces the amount of data by trying to reduce words to their stem (for example: tokens => token, sleeping => sleep, and so forth).

4. The **normalizer** is called to normalize words. Depending on the notes.ini setting (DS_LOWERCASE_KMAP), the wrapper converts all the remaining stemmed tokens to lowercase (if the variable is set to 1) and returns to the calling spider. For example, *Discovery* at the beginning of a sentence will become *discovery*, but *I* will not be normalized to *i* but remain *I*. The reason for putting all words in lowercase is that this makes token counts more accurate: *Discovery* and *discovery* would be counted as two different words without this step.

5. The InXight tool **tagger** identifies types of tokens. Only nouns are identified.

6. Tokens are weighted. Tokens occurring in titles are counted 3 times, tokens occurring in subject fields are counted 3 times, and tokens occurring in KeyFields are counted 2 times within their respective token count.

7. Every token is assigned a unique ID.

Following are the files that are written by the InXight tools in the ongoing process of K-map building:

- **InXight_cnt** - Count of tokens per document (records document within cluster). Only document “weighted” token counts are stored in this file. Occurrences are computed dynamically by the K-map Building task from the document information stored in this file.
Label_InXight_tok - Count of tokens (for labels, nouns only) per document within the K-map cluster. Occurrences are computed dynamically from the label_inxight_cnt data.

InXight_tok - Contains individual tokens and the number of occurrences within a K-map cluster.

ATGRoot DocID - Contains the DB2 document ID string for each of the documents whose content and label token information is stored in inxight_cnt and label_inxight_cnt files respectively.

ATGRoot DocInfo- Maintains state information about each document processed: whether it is included in the initial training set, if it has been categorized yet, or read only by the K-map Building task.

Clustering via IBM SABIO Tools
The K-map building (clustering) is done by the IBM SABIO Tools for document clustering, which use the technique of vector-categorization to place all documents (vectors) into an n-dimensional vector space, trying to find clusters (categories) these documents could fit to. These clusters can overlap each other or consist of a number of smaller clusters (subcategories).


Document abstracts via InXight tools
Discovery Server leverages the InXight tools SDK for various tasks when building or maintaining the K-map. One part of this set of tools is used to create automatic document abstracts. It is called InXight summarizer.

From a very high level point of view, InXight summarizer weighs sentences in a document and picks the sentences that scored best out of all sentences in that document, until a maximum length for the abstract is reached. But what is the criteria summarizer uses to weigh sentences?

One of the most important files for the summarizer is in the lotus\DS\inxight\lx-2\lang folder of your Discovery Server installation. It is called <language>-std.feature-config, where <language> stands for one of the supported languages.
In this file you will find various terms that are used to support summarization. The summarizer checks for the following:

- Is there already an Abstract in the document? If the summarizer finds words in the `<language>-std.feature-config` that indicate an abstract, the sentences in this abstract receive a high relevance score.
- If a sentence contains conclusion phrases, this sentence receives a high relevance score.
- If a sentence contains so called drop phrases, its relevance score is reduced.
- If a sentence contains words that are contained in a user query, this sentence receives a higher relevance score.
- Sentences in the first and last few paragraphs receive a higher relevance score.
- Sentence fragments (no noun or no verb contained) receive a reduced relevance score.
- The title of a document receives extra weight.
- The most frequent content words are defined as thematic words. If a sentence contains thematic words, its weight is raised. (However, extremely frequent grammatical words, like the, is, and so forth, are ignored as potential thematic words.
- Long sentences are preferred over short ones.

**Discovery Server procedure for K-map building**

The following procedure is utilized by the K-map building task when actually processing and building the K-map

1. All words in the XML documents are counted.
2. Certain words are selected; these are called cluster terms.
   a. This selection is based on a value called term-discrimination value, which is used to balance the size of all clusters, making them not too big and not too small.
   b. Selecting the cluster terms ensures that the vector space is not too wide or too narrow to find appropriate clusters. This has to be done to put the clusters in one single cluster tree, called the taxonomy. The cluster terms represent dimensions in a multi-dimensional space (each cluster term represents one dimension), to be used when placing documents in this vector space.
3. Documents are placed in the vector space. The more frequent cluster terms have been counted; the placement of this document in the dimension of a cluster term is determined by this value. This is done for every cluster term
(dimension) until the document can be placed into the vector space as a point or vector (from the origin of the vector space to the location in the vector space where a document has been calculated to be placed)

4. Documents or vectors that are near each other are defined to belong to one cluster.

5. This cluster is named after the cluster terms found in most documents in this cluster.

Cluster terms found in fewer documents are not selected to label a cluster (even though the documents containing these particular cluster terms still belong to this cluster).

Clusters that are found near each other in this multi-dimensional vector-space are clustered into one cluster. This cluster is named after the most frequently used words in the documents the cluster contains. This process runs until there is only one cluster left over, representing the root of the taxonomy tree.

### 2.7.2 Categorizing documents

Once the initial set of clusters has been created, the K-map building service (ATG) provides an automatic classifier. Through this service, the K-map building service compares the words in new documents (and documents in the Uncategorized section of the K-map) to the words in the clusters it has already created.

If the new documents are similar to the documents already in existing categories, the new documents will appear in the same categories. If the new documents are not similar (that is, they do not use cluster terms similar to those in documents already clustered into categories), ATG puts them in an Uncategorized Documents category. This uncategorized category must then be evaluated by the human editor as part of the process of K-map refinement.

Categorization is enabled by clicking **Categorize** in the Discovery Server Control Center.

### 2.7.3 K-map searching

Search queries submitted to the K-map information retrieval system are done using full text queries, while the full text index covers categories, documents, and people information. While categories and document indexes are stored in the core eight-segment Full-Text Index, people information is stored in a separate index. If Discovery Server is integrated with K-station or WebSphere Portal Server 2.1, there is another index created, which is called LPPlace.ft. This additional index holds the indexes of place content.
In the case of the K-map of Discovery Server, since DB2 stores all the documents the taxonomy consists of, this is the fastest way to search for content.

To make the K-map browsing UI faster, there is a caching mechanism with a configurable expiration time, caching all the information that has been queried from the DB2 backend databases.

Per default settings, this cache is refreshed every 60 minutes, and is populated when the first query on the K-map is submitted by any user. The notes.ini settings to change the presets for the K-map cache are:

- LDS_CURSOR_MEMORY is the maximum amount of data to cache in memory in megabytes; the default is 10 megabytes
- LDS_CURSOR_DISK is the maximum amount of data to spill to disk in megabytes; the default is 100 megabytes
- LDS_CURSOR_TIME is the time in minutes to hold data in the cache; the default is 60 minutes

When the cache reaches a size of 110 MB (when the default setting of 100 MB cache size is used), the least recently used cache entry is expired. When LDS_CURSOR_TIME in minutes has passed since the last cache update, the cache is flushed.

**Note:** When users perform a search, search results are filtered on which documents a particular user is allowed to see by using the ACL information defined in the data repositories. All ACL information is stored in the K-map (DB2).

Discovery Server uses the back-end features identified in Table 2-3 when searching for content.

**Table 2-3  Features used by each search type**

<table>
<thead>
<tr>
<th>Search type</th>
<th>Feature used</th>
</tr>
</thead>
<tbody>
<tr>
<td>Categories about</td>
<td>➤ Phrase matching (match entire string exactly, even if unquoted)</td>
</tr>
<tr>
<td></td>
<td>➤ Fuzzy search (match alternate spellings)</td>
</tr>
<tr>
<td></td>
<td>➤ Partial-word matching (match word containing search term)</td>
</tr>
<tr>
<td></td>
<td>➤ Stemming (match part of string)</td>
</tr>
<tr>
<td></td>
<td>➤ Thesaurus (match alternative words for query string)</td>
</tr>
</tbody>
</table>
Users can use the operators identified in Table 2-4 to search in the K-map.

**Table 2-4  Valid K-map search operators**

<table>
<thead>
<tr>
<th>Operator/expression</th>
<th>Example</th>
<th>Meaning</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCRUE</td>
<td><em>,,</em></td>
<td>Cat, Dog, Fish</td>
<td>Weighted OR</td>
</tr>
<tr>
<td>AND</td>
<td>&quot;&amp;&quot;</td>
<td>Cat AND Dog</td>
<td>Boolean AND</td>
</tr>
<tr>
<td>OR</td>
<td>&quot;</td>
<td>&quot;</td>
<td>Cat OR Dog</td>
</tr>
<tr>
<td>NOT, &quot;!&quot;</td>
<td>Cat AND NOT Dog</td>
<td>Boolean NOT (Exclusion)</td>
<td>Need AND/OR before NOT</td>
</tr>
<tr>
<td>*,.</td>
<td>Cat -Dog</td>
<td>Exclusion</td>
<td></td>
</tr>
<tr>
<td>&quot;+&quot;</td>
<td>+CAT +DOG</td>
<td>Requirement</td>
<td></td>
</tr>
<tr>
<td>Operator/ expression</td>
<td>Example</td>
<td>Meaning</td>
<td>Comments</td>
</tr>
<tr>
<td>----------------------</td>
<td>---------</td>
<td>---------</td>
<td>----------</td>
</tr>
<tr>
<td>PARAGRAPH</td>
<td>Cat PARAGRAPH Dog</td>
<td>Paragraph Proximity</td>
<td></td>
</tr>
<tr>
<td>SENTENCE</td>
<td>Cat SENTENCE Dog</td>
<td>Sentence Proximity</td>
<td></td>
</tr>
<tr>
<td>TERMWEIGHT</td>
<td>TERMWEIGHT 80 Cat AND TERMWEIGHT 30 Dog</td>
<td>Weighted Search</td>
<td>Need AND/OR before subsequent TERMWEIGHTs</td>
</tr>
<tr>
<td>FUZZY</td>
<td>FUZZY 66 Cat OR FUZZY 80 Dog</td>
<td>Weighted Fuzzy Search</td>
<td>Need AND/OR before subsequent FUZZYs</td>
</tr>
<tr>
<td>NEAR</td>
<td>Cat NEAR Dog</td>
<td>Proximity</td>
<td></td>
</tr>
<tr>
<td>EXACTCASE</td>
<td>EXACTCASE &quot;dog&quot;</td>
<td>case sensitive search</td>
<td></td>
</tr>
<tr>
<td>&quot;*&quot;</td>
<td>&quot;?&quot;</td>
<td>Dog* OR Cat?</td>
<td>Wildcard</td>
</tr>
<tr>
<td>&quot;&quot;</td>
<td>&quot;This is a short phrase&quot;</td>
<td>Text phrase search</td>
<td></td>
</tr>
<tr>
<td>ACCRUE with Proximity</td>
<td>Cat ACCRUE Dog Near Water</td>
<td>Find documents that contain both &quot;Cat and Dog, then find Water near Dog</td>
<td></td>
</tr>
<tr>
<td>ANDs with ORs and NOTs</td>
<td>Cat AND ! Dog OR Fish</td>
<td>Find documents that contain &quot;Cat&quot; but not &quot;Dog&quot; or contain &quot;Fish&quot;</td>
<td></td>
</tr>
<tr>
<td>Weighted search and ORs</td>
<td>TERMWEIGHT 20 Dog OR TERMWEIGHT &quot;Fish&quot; OR TERMWEIGHT 80 Cat</td>
<td>Find documents with either &quot;Dog&quot; or &quot;Fish&quot; or &quot;Cat&quot;, rank them by the sum of their term weights</td>
<td></td>
</tr>
<tr>
<td>ORs and NOTs</td>
<td>Dog OR ! Fish</td>
<td>Find documents that have &quot;Dog&quot; or don't have &quot;Fish&quot;, rank those highest for which both conditions are true</td>
<td></td>
</tr>
</tbody>
</table>

**Tip:** Per default setting, wildcard search is disabled to improve performance. If you want to enable wildcard search for the Discovery Server user regardless of whether they issue searches with or without wildcard search operator *, add KmapServlet_AddSearchWildcards=1 to your notes.ini file.
2.7.4 K-map indexing

In order to find all documents clustered within the created taxonomy, Discovery Server utilizes the K-map indexer, indexing all XML documents put onto the LDSFullTextWorkQ.queue. The FullText engine used by Discovery Server is the IBM GTR (Global Text Retrieval) Version 4.1, the same engine used in Notes/Domino.

When a document is indexed by Discovery Server, it indexes all contents of the documents. The index information is broken out by repository into different indexes. The Discovery Server Fulltextindex server has one logical index, which is made up of eight separate physical index segments. This number is chosen because it allows faster indexing. The index files are in the file system under \\Lotus\DS\Data\ftdomain.di\Discovery Server in eight separate directories named LDSIDX00 through LDSIDX07. Person profiles and K-station or WebSphere Portal Server places are indexed into a separate Fulltextindex, which also resides on the Fulltextindexing Server.

**Tip:** We recommend that you run KMapIndexing on a dedicated, secondary Discovery Server to make Indexing perform better.
2.8 Multi-language support

There are several key components of Discovery Server that have to treat data repositories language-dependently. These language-dependant components are:

- ATG Scan task
- InXight tools Stemmer
- InXight tools Tagger
- K-map search feature

The ATG Scan task removes words from the XML document stream which can be found in one of the stopwords.txt files, one file for each language supported. When editing the stopwords files, make sure there are absolutely no duplicate words in the list. Usually you will add words here that you don’t want the classifier to create categories (clusters) with.

The InXight tools Stemmer reduces the words found in the XML document stream to their stem, for example, reducing driving to drive, cars to car, and so forth; while the InXightTools Tagger eliminates language-specific parts of speech.

The InXight tools require a set of special files for every language to be supported, which is deployed with the installation of Discovery Server. These files usually do not need to be customized and are ready to be used without editing.

Performance tuning for a single language

If you plan to use your Discovery Server with documents that use only one language, and you do not plan to spider documents in any other language, you can increase spidering speed and K-map quality by modifying the file std.langid-config in your InXight directory with a plain text editor like notepad. Remove languages you don’t need under the tag <encodings-languages-covered> <list key = "utf_16">. Do not touch any other settings than those under this tag. Make a backup of this file before editing.

If you need to edit the InXight word files themselves, refer to “Document abstracts via InXight tools” on page 44.

2.9 Thesaurus

A thesaurus tool based on the IBM LanguageWare thesaurus is shipped with Discovery Server 2.0. This tool is used to define wordlists that act as synonyms for any given word in the search section of Discovery Server.
The implementation that ships with Discovery Server only supplies English language terms, and also supports only ASCII text, so it provides only a limited set of capabilities. Administrators may follow the instructions included within the readme.chm file supplied with Discovery Server to augment the Discovery Server-supplied thesaurus with new terms.

For example, if your company is in the chemical industry, and your users search for the term H2O (the chemical formula for pure water), they will want to find documents that either contain the term H2O or the word water. You can achieve this by adding H2O as a synonym for water in the thesaurus of Discovery Server.

There are two major thesaurus components that are installed during the Discovery Server installation process:

1. The naddenda utility tool is a command line tool that is used to load the Discovery Server-supplied thesaurus with words and synonyms. Details on command line tool usage can be found in the readme.chm file.

   **Tip:** We recommend that you add all words as synonyms to the thesaurus in lowercase letters. This will enable you to find all lowercase, all uppercase and first letter uppercase/rest lowercase occurrences of a word when submitting a query.

2. The SPI (service provider interface) component is the actual thesaurus used to look up words when you use the Discovery Server search capabilities. This file is called ntheslang.dll, and it can be replaced by custom/third party applications for those who want to replace the LanguageWare capabilities. For example, corporations may have an existing thesaurus solution that they wish to leverage for use with Discovery Server. This existing thesaurus would need to be wrapped in a custom application/wrapper that implements the Discovery Server thesaurus SPI.

   The Discovery Server thesaurus SPI is based on a C++ interface; details on its use can be found in the readme.chm file.

### 2.10 Other architectural aspects

There are a few aspects of the Discovery Server architecture that differ from the general architecture described in 2.1, “The external view of the Lotus Discovery Server” on page 14. For example, the K-map editor is the only user interface component supplied with Discovery Server that talks directly to the Discovery Server backend services via CORBA/IIOP. All other interfaces are more pure HTML/XML/DHTML, and talk through the intermediate servlets and other API components to access backend services.
These additional architectural features are described in the following sections.

2.10.1 K-map editor architecture

The K-map editor is a stand-alone application, which can be installed on any client machine in your network. Taxonomy editors simply click the following URL and follow the instructions at this URL to install the editor on their local machines:

http://primaryserver.domain.com/kmapeditorinstall.html

The user ID that is used to authenticate against the Primary Discovery Server should be a "flat" common name user id (for example, john smith instead of john smith/domain); and the connection should be made to the fully qualified name of your primary Discovery server (for example, server.domain.com). The user should also be member of the TaxonomyEditors Group. Additionally, you may want to add your Discovery Server Administrators group as Taxonomy Editors.

All edits a K-map editor client performs to the K-map are communicated to Discovery Server's KMapEditor Service using the CORBA IIOP protocol.

**CORBA and IOR**

There are many different CORBA implementations available on the market, but they all rely on a location server providing the information on how to reference a given CORBA object. The reference/location service in Discovery Server uses the Interoperable Object Reference (IOR) object to provide the reference.

The IOR provides the following information:

- **IIOP version**: The IIOP version this ORB implements
- **Host**: This ORB's host machine TCP/IP address
- **Port**: TCP/IP port number where the ORB is listening for client requests
- **Key**: Value uniquely identifies the servant to the ORB exporting the servant
- **Components**: A sequence containing additional information applicable to object method invocations, such as supported ORB services and proprietary protocol support

The specific IOR reference involed in the K-map editor is discussed in the next section.

**K-map editor communication sequence**

The K-map editor uses the following message sequence to find the CORBA-based K-map editing service on the server running ATG:
1. Send HTTP request to the NameService servlet to retrieve information on where to find the KMapEditor service.

2. The NameService servlet returns an IOR string (in hex encoding).

3. KMapEditor connects to the KMapEditor service on ATG server using IIOP Protokoll (Port 5051).

4. All subsequent communication between KMapEditor and KMapEditor service uses DIIOP protocol.

**Attention:** If the K-map editor is having problems connecting to Discovery Server, this could be for two reasons:

1. Your company uses DNS aliases, which, when retrieved by the K-map editor in the IOR string, cannot be resolved by the client machine properly.

2. Your Discovery Server machine uses two network interface cards and the IOR returns an unresolvable host name or IP address. Use the notes.ini setting LDS_Ior_Host to specify the correct IP address to be used by the IOR.

The K-map editor application establishes semi-direct communication with the K-map and ATG's direct work queue. It uses the CORBA KMapEditor service to write to the ATG direct work queue and DB2. When you start K-map editor, you might get a message saying that “The K-map Building service is currently updating the category tree...”. This tells you that the editor is trying to get exclusive access to the K-map so the server isn't changing the K-map while you're trying to edit it.

This *locking* mechanism does not lock out other K-map editors. When you start the K-map editor and connect to the K-map, the server stops the K-map building task, in essence stopping the server from modifying the K-map while you're editing. The maximum time ATG can use to finish its work when a K-map editor is requesting the lock on the K-map is predefined to 45 seconds. The server will automatically resume managing the K-map once you close your editing session.
The K-map must keep ATG, Metrics and FullText informed whenever any changes are applied to it. Some operations performed by the taxonomy editor need to be communicated quickly to ATG for the following reasons:

- They affect all pending messages in the normal ATG work queue. For example, if the ATG work queue has 1000 messages and an editor modifies the taxonomy, we want the editor's modification to happen before the 1000 pending messages are processed. If the editor's request happens later, then some of the 1000 messages might need to be processed again.

- Some operations, such as subdivide and categorizing uncategorized documents, need to happen as soon as possible. This is because an editor is waiting to review the result. We understand that these actions are not immediate, but since an editor is waiting, they should occur before other processing.

This is accomplished by the K-map and ATG using a separate work queue. ATG will always check this work queue before reading a message from the normal work queue.

Figure 2-10  Taxonomy editor K-map interaction
By communicating directly with the K-map editor service, all edits you make happen immediately. If you went through a work queue, you wouldn't see any changes until the server processed the requests in that queue, which would make it hard to track the changes you make during that editing session.

### 2.10.2 Sametime integration

In the standard K-map UI that comes with Discovery Server, a special Java applet that is locally stored on the users workstation (the Discovery Server People Awareness Client, or DSPAC applet) provides the Sametime awareness functionality. This applet is coded using a Discovery Server-specific interface to Sametime that provides the authentication and data retrieval required to integrate.

If you develop your custom UI with the Discovery Server API toolkit, we recommend you use the new functionality of Sametime 3.0, called **Sametime Links**, which provides other features and does not need the DSPAC applet provided with Discovery Server. The ST Links toolkit is described in more detail in 8.5.3, “IBM Lotus Sametime” on page 249.

Three databases involved in the custom Sametime integration process are shipped with Discovery Server; they are:

1. **NAB: Names.nsf** - This database is used to resolve the Sametime server's fully qualified Internet hostname for a particular user.

   **Tip:** If you are using an LDAP directory instead of a Domino directory, you will need to add a value to your LDAP schema so that Discovery Server can resolve the address for the Sametime server. The value to add is called **SametimeServer**.

2. **Sametime Secrets: stauths.nsf** - This database is used to get the secret key to generate a valid token for a particular user at a particular time.

3. **DS Main DB: dsmain.nsf** - This database contains forms and agents that are used to retrieve data from the above databases and generate a unique token that the People Awareness Component (PAC) of a particular user uses to log on to a Sametime server.
Figure 2-11 illustrates the process of logging a user on to Sametime. The steps it shows are as follows:

1. When the Discovery Server Sametime PAC (DSPAC) applet is initialized, it asks the agents in dsmain.nsf for the hostname of the user’s associated Sametime server and a valid token to log into Sametime.

2. The agents retrieve the user's Sametime server's hostname from the person document in names.nsf, or the SametimeServer field in the LDAP directory used by Discovery Server. The agent also retrieves a valid secret key from the stauths.nsf database, which is used to generate an individual token for this user to log in to Sametime. These two pieces of information are then passed back to the Discovery Server DSPAC applet.

3. The DSPAC applet logs the user on to sametime using the token.

4. Sametime uses the same secret key to validate the token.
Planning for a successful deployment

The deployment of any product is made easier through careful consideration of relevant issues and thorough planning before roll-out. By taking time to consider how the product is going to be used, analyzing where you currently stand, and planning how you are going to get there, you can make a major contribution towards a successful roll-out of the Lotus Discovery Server within your organization.

This chapter is intended for anyone who has the task of deploying a Lotus Discovery Server system in their organization.

It includes advice on:
- Determining how to leverage Discovery Server to solve real business needs
- The importance of analyzing and selecting your data
- Establishing a clear project plan
- Building an effective Discovery Server deployment team
- General strategies which can help you achieve a successful deployment
3.1 Identifying the problem to solve

If your organization has already purchased Discovery Server, then hopefully it has been purchased with a specific problem in mind. That is, the technology has not simply been purchased because someone thought it was a cool piece of technology. Deployments of new technologies in any organization often fail for this reason alone – that they are simply technology for technology’s sake. Technology must be planned, architected, and deployed with a real business problem to solve, or the odds of success are poor.

3.1.1 Knowledge management doesn’t have to be scary

Overall, Discovery Server plays a key part in any company’s knowledge management strategy. Should you already have a clearly defined and articulated KM strategy, then you will probably see right where and how Discovery Server fits in, and might want to skip to 3.2 “Create a project plan” on page 66.

However, if your organization is new to the knowledge management area, don’t be scared away. The term “knowledge management” often leads one to begin thinking about long-term strategies, expensive software, and Chief Knowledge Officers; but you do not need a full knowledge management strategy/plan/vision to successfully use and utilize Discovery Server. In fact, leveraging the basic capabilities of Discovery Server to solve a clear business problem now, may let your determine how various knowledge capabilities can benefit your organization in larger scale ways, and actually help kick-start future organization-wide knowledge management efforts.

If you fit into this “new to KM” category, then the rest of this section is for you. We give you some practical steps and guidelines for determining the “knowledge/information maturity” of your organization, and its readiness for leveraging Discovery Server technology.

3.1.2 Finding an information problem

In general, Discovery Server is geared at solving problems that are of an “informational” nature. These are the kinds of problems where people are wondering:

- “Does anybody already know this?”
- “Who can help me with this?”
- “Somebody must have done this before.”
- “If I could just find some details on this!”
If you hear these kinds of questions, and the lack of answers to these questions is limiting the ability of others to work effectively, then chances are you have found a problem which Discovery Server can solve.

Some example information-related solutions that can be enabled by leveraging Discovery Server are:

- Arming your sales force with immediate product information from various disparate systems, as well as access to product knowledgeable people from across the organization, to decrease the time required to close a deal.
- Enabling your project managers to quickly locate past project documentation to jump-start a new project, as well as people with the right set of skills to staff the new project.
- Enabling your customer service representatives to quickly find the information they need to answer a customers question while on the phone, by either quickly finding relevant documents or instant “chatting" with relevant experts.
- Enabling your researchers to quickly locate others who may be working with similar technologies or concepts in other parts of the world so that efforts are not duplicated and ideas can be shared.

Additionally, when identifying the problem, be sure to:

- Target functional areas that have a clear, focused business objective with well-defined benefits.
- Avoid multi-functional areas that have multiple objectives and offer intangible benefits; such areas can easily scuttle your Discovery Server rollout through lack of progress and unproductive effort.

### 3.1.3 Perform a knowledge/information audit

Once you have clearly defined the problem, a knowledge or information audit lets you determine exactly what data repositories you company has available that can help with the problem, and the state of those repositories with respect to their ability to be leveraged by Discovery Server. The following questions will help you to focus on the existing knowledge in your organization.

**What information assets do you currently have?**

You need to clearly identify which information repositories have value to your organization. Are there central repositories used to post information within a larger taxonomy, or are your data repositories more departmentalized? There may be more formal repositories such as Lotus Notes or corporate intranets, or more informal repositories such as a “products” directory on the corporate file system. Any of these types of repositories can be leveraged by Discovery Server out-of-the-box.
Are there other proprietary repositories of data stored within home-grown systems, or ERP/CRM systems? These types of repositories can also be leveraged by Discovery Server, but do require additional work via use of the XML Spider described in 8.3 “Data source integration via the XML spider” on page 217.

**What state is your current data in?**

For each of the data repositories identified, how current, categorized, and secure is the data? Consider the following:

- Does the data appear regularly updated/added to? If not, the repository may still be valuable for including in your Discovery Server index/taxonomy; but you may need to search for another data repository to include that may have replaced this one.

- What kind of data is held in the repository? In general, repositories which contain only highly graphical documents (bitmaps, gifs, jps, and so forth), or highly numbers-oriented documents (such as spreadsheets), should be avoided because they will not categorize well if there is not enough accompanying text to describe the documents.

- Is all the data important within the repository? It is possible that only a subset of the data held within the given repository will provide value. If this is the case, this should be recorded so that no unnecessary effort is spent processing data that is of no value to your problem at hand.

- Do sensitive documents in the repository have clearly identified and applied security controls? Discovery server will honor any security that exists at the repository level to protect sensitive documents. If such documents are not secured, but you wish to make a given repository available within Discovery Server, then you should apply the security at the repository level.

- Does the repository include any sort of metadata that describes each document in the repository in terms of authors, titles, and categories? Discovery Server does not need this information to function and provide a basic level of value, but this type of information will become more crucial as you use the more advanced capabilities of Discovery Server.
  - Category types of metadata are key to ensuring that the taxonomy generated by the K-map builder (the automatic taxonomy generation) requires less manual editing.
  - Author types of metadata are important because this identification enables the discovery and assignment of expertise/affinities.
  - Document titles are also crucial since, without good titles, end users will not be able to distinguish one document from another.
If you have a data repository with minimal metadata, but which you feel is key to helping solve your business problem, then adding some steps into the deployment process to add or improve the metadata is recommended.

Overall, be sure to review 6.3.1 “Choosing repositories” on page 161 for more details on selecting and identifying good data repositories.

**How much information is stored in user e-mail files?**

Some organizations may have large amounts of corporate knowledge stored in the e-mail files of users. This may be especially true for organizations that have not had a formal knowledge management strategy that includes policies encouraging employees to share data/documents, as well as places for employees to put such data. In such cases, users will often save key pieces of knowledge within their e-mail files.

If a large amount of data and knowledge exists within e-mail files, then a large amount of knowledge exists within your employees themselves (that is, tacit knowledge). It is important to note that data in user e-mail files is not exposed to other users directly within a Discovery Server deployment. Discovery Server instead uses the data in e-mail files to identify employee “affinities” to certain topics, and can then make these affinities available to all employees to discover via a simple search.

See 7.3.3 “The use of e-mail in generating affinities” on page 198 for more details on the manner in which Discovery Server can leverage e-mail knowledge to help you identify people expertise.

Due to the sensitive nature of accessing employee e-mail, it is recommended that you first introduce the users to Discovery Server and the concept of affinities based on more public documents, before approaching the spidering of e-mail.

### 3.1.4 Examine your organization’s knowledge culture

Any knowledge management effort requires more than just technology. KM initiatives often require fundamental changes in the way an organization, or even a specific department, uses and shares information. The importance of considering the cultural aspects of any Discovery Server deployment cannot be stressed enough. Consider the following questions to assess your organization's knowledge culture.

**Who owns the information and will they allow access to it?**

Some organizations have very stringent information or security controls on their data. They may only allow sales people access to the basic sales information, developers or engineers access to product specs, and so forth. If this is the case
with your identified data, then some cultural/organizational changes may be required for you to fully utilize Discovery Server.

Knowledge management is not about maintaining clean and polished databases. It is about encouraging a workplace where people can ask the “who” and “where” kinds of questions. This requires open systems and management support for inter-departmental data sharing and interaction.

Are there employee incentives to share knowledge?
One other key consideration when beginning a deployment of Discovery Server is analyzing your organization's culture towards the sharing of information. Discovery server depends on having access to quality and current knowledge so that users get useful results. Are your employees encouraged in some manner to post their key documents to a publicly accessible data source, or post troubleshooting steps they discover, and so forth?

Additionally, are your employees encouraged to help each other out, even if this means spending some time assisting another employee in a different department? Discovery Server will help you identify the expertise in your organization, but your organization's culture and incentive systems determines whether those “experts” will actually help others!

3.1.5 Discovery Server can be used in more than one way
In many cases, organizations will find that deploying Discovery Server in a phased manner is the best approach to getting the most immediate benefit and results out of the product. The business problem being tackled may be solved via the introduction of Discovery Server's basic search capabilities, which require the least effort to deploy. Alternatively, the problem defined may require the full taxonomy browsing and affinity generation capabilities of Discovery Server; but either way, starting with the basic search capabilities will allow your users to get more comfortable with Discovery Server concepts, while also allowing the deployment team to fine tune and hone the advanced Discovery Server capabilities – and most importantly, allow management to see immediate results from their investment.

Some common “phases” or options for a Discovery Server deployment could be as follows:

Basic search
After following some of the other steps described in this section to clearly identify and prepare your data sources, deploy a basic search interface to Discovery Server only. This will allow users to search the full text index generated by Discovery Server as it spiders your data repositories. Discovery Server search
capabilities will often be a natural extension to existing search capabilities deployed in your organization.

Such a basic search interface may be created using API samples provided with the product and discussed in 8.1.1 “A simple Discovery Server search interface” on page 208.

**Search and expertise**

Once the search capabilities have been tested and deployed successfully, you can enable basic affinity generation capabilities on a small subset of “experts.” This will allow users to then receive both documents and people (experts) in their search results. You can use the automatically generated taxonomy during this stage, since your search results interface does not have to expose the categories, or taxonomy, associated behind the affinities.

You will have to spend some time defining and building a taxonomy for this step. While the taxonomy need not be presented to your regular end users, it will need to be understandable to the handful of experts you initially enable affinity generation for.

Such an interface is not provided out of the box, but is not difficult to build based on the API samples.

** Taxonomy browsing**

After your users are comfortable with the basic search, and the concept of experts based on affinities, you can begin exposing them to the full taxonomy browsing capabilities within the provided K-map client interface (should this step even be required to solve your identified business problem).

You will probably want to proceed to this step only after some additional time has been spent defining, building, and fine tuning your taxonomy (as described in Chapter 6, “Developing and maintaining an enterprise taxonomy” on page 157), since at this point the taxonomy must make sense not to just a core group of experts, but to all of your end users.

**Custom and alternative interfaces**

You may also want to consider alternative interfaces at this time, to further leverage your Discovery Server deployment to meet your business needs. For example, you might deploy the Palm PDA-based Discovery Server Everyplace interface to allow your sales people access to information to close a sale while on the road. See 8.5 “Complementary products” on page 244 for more information on Discovery Server Everyplace, and other complimentary products to consider when defining a Discovery Server-based solution.
3.1.6 Finalize your goals

After you have found an information-related problem, analyzed the data available to solve it, analyzed the cultural barriers, and considered different options available from Discovery Server; it is time to clearly document and define the goals for your Discovery server deployment. The goals should be as clear and descriptive as possible, including clearly defined measurements that can be used to evaluate the success of the deployment.

This is an important step to ensure that all those involved with the project clearly understand where you are going and what problems you are solving. Additionally, the goals that you define play a key part in helping you establish your overall detailed project plan.

3.2 Create a project plan

After clearly identifying the problem and defining your goals, you are now ready to create a project plan that defines how the project team will deploy the Discovery Server. An important piece of advice at this stage is: “Be realistic in deciding what you can achieve with the resources and time you have available.”

The following checklist highlights some of the issues to consider when creating your project plan. Remember to include site-specific issues (for example, your company’s IT security policy).

- Always keep the goals and scope of the project in mind.
- Break up (or downsize) the project into manageable segments.
- Match the resources available to the size and complexity of the project.
- Determine what size pilot you are capable of supporting with the resources available.
- Include education and promotion as part of your project plan. This is key to addressing any cultural barriers you identified when determining the project scope and goals.
- Allow for production of status/management reports.
- Identify critical stages and high-risk factors in the project and propose how they will be managed.
- Create timetables that can be used to monitor project status.
- Identify key players and define communication channels between them.

Following is a proposed framework for creating a Discovery Server deployment plan. This can be used as a starting point for drawing up your own project plan.
1. **Prepare a business case**
While you may have clearly defined a problem, a solution using Discovery Server, and documented project goals/scope; you may need to spend additional time providing business justification to management. Such a “business case” document seeks approval for, and commitment from, management to support your proposal. This is usually where estimates for resources (staff, hardware, training, promotion, and so forth) are submitted, and where approval for funding is sought from management.

2. **Assemble a multi-disciplinary team**
Knowledge management initiatives usually involve many areas of your organization. Implementation of Discovery Server often requires more of a commitment than rolling out a simple desktop application. Planning a deployment can bring into play a range of organizational and technical issues which involve training, promotion, system administration and design, research, and so on. Discovery Server deployments, like other enterprise-wide systems, thus require the task of establishing a team with the necessary skills from existing staff resources. This is often managed as a matrixed team, where team members continue their normal jobs, while also participating in the deployment efforts.

3. **Build a test/demonstration environment**
This stage achieves the following objectives:
- Enables team members to become familiar with the product
- Allows proof-of-concept tests to be conducted
- Allows the product to be demonstrated to key players such as line-of-business managers
- Helps identify technical, cultural, and organizational issues not previously anticipated

4. **Prepare for a pilot study**
This stage defines the parameters for conducting a pilot study of your solution. A pilot of any new technology is crucial to determine that your defined project goals and scope are feasible and will solve the problem as expected. A successful pilot will allow you to work out the kinks in your solution/plan – prior to any issues having a larger scale impact.

The pilot scope, people, data, dates, support arrangements, training requirements, and promotion material are documented in a detailed pilot project plan. Critical success factors for a pilot are support from management, clearly defined objectives, careful allocation and management of resources, selection of a representative user base, and capture of meaningful information for analysis.
5. Conduct the pilot
Make sure you have appropriate recording mechanisms in place for measuring, monitoring, and managing the pilot. Be prepared to modify or suspend the pilot if the project outcomes are not being realized or significant problems arise. Remember: The pilot is (normally) meant to be a trial run for a full-scale deployment.

6. Conduct a postmortem/debrief on the pilot
Consider the following questions:
- Was the pilot a success?
- Did it provide sufficient information to confirm you are ready for a full-scale deployment?
- What did you learn?
- Are your resource estimates (people, hardware, training material) for a full deployment correct?

7. Prepare for a production release
Integrate lessons learned from the pilot into the final rollout plan. Acquire and put in place infrastructure to support a full-scale rollout of your Discovery Server-based solution. Adjust server sizing estimates to meet projected load. Commit timetables for production release and advertise the implementation schedule. Conduct training and corporate awareness seminars to create the desired mindset and promote shifts in organizational and cultural behavior. Update the project plan for a full-scale deployment.

8. Production rollout
Make sure the support infrastructure is ready to accept the challenge. Post regular updates on the progress and uptake of the product. Manage problems and respond to feedback on implementation and usage of the product.

9. Post-implementation report
The final stage involves conducting an honest assessment of the success of the rollout. This may involve running a survey to collect end-user feedback and interviewing line-of-business managers to ascertain whether anticipated benefits have been realized. Bring the rollout team together for a post-rollout analysis and assessment of the following:
- Were the expected benefits realized following the rollout of the Discovery Server?
- What remains to be done?
- Have desirable shifts in cultural and organizational behavior been achieved?
3.3 Assemble the solution team

Planning and rolling out a solution using the Discovery Server can require considerable consultation and collaborative effort between technicians, line-of-business managers, end users, data custodians, librarians/taxonomists, and the executive. Lotus recommends establishing a multi-disciplinary team which includes representatives from various areas and levels of responsibility to guide a successful deployment. A team built on such a foundation will be better equipped to deal with the technological, organizational, and cultural issues that come into play when deploying knowledge management technologies.

3.3.1 Composition of a Discovery Server deployment team

Lotus has identified a number of roles within a typical Discovery Server implementation that describe the type of work people in the organization will be performing. In looking at these roles, you can identify attributes and skills that would be useful for team members to possess. Keep in mind the size of your team is likely to grow as you progress through various stages of implementation: from initial proof of concept and demonstration trials through to conducting a pilot, and eventually, final deployment. Team members may be able to cover multiple roles depending on the skills they possess and the number of people available.

Tip: Sharing roles between team members encourages sharing of information and reduces the impact of staff turnover on the remaining team members.

Sponsor
This person could well be the “Chief Knowledge Officer” if you are deploying a large-scale Discovery Server solution, or at least someone who is involved at a management level with responsibility for the problem which the Discovery Server-based solution will solve. The person is likely to be someone who has been with the company for a number of years, knows the organization well, and can act as a “visionary” who is able to articulate the solution strategy. In brief, the sponsor oversees the entire Discovery Server project, but is probably not directly attached to the team.

As part of this role, the sponsor is usually responsible for:

► Establishing the scope of the deployment
► Managing the deployment and administration of team effort
► Keeping executives informed on the status
► Providing strategic advice
Giving input to funding and allocating team resources
Selecting team members

General skills which are assumed or required include that the sponsor:

- Understands the value of information to the business the company is in
- Can influence senior management and has respect within the organization
- Understands the company culture
- Possesses interdisciplinary knowledge (knows who does what within the company)
- Possesses overall knowledge of the type of data/data sources available
- Possesses overall knowledge of data security and data privacy policies
- Knows who to talk to in order to get information

**Metrics evaluator**

This person uses metrics to assess what (and how) information flows into, out of, and within the corporation. Within the Lotus Discovery Server deployment team, the role of this person will change depending on what stage the deployment is in. Initially, this person will be concerned with validation and analysis of metrics data to ensure usable information is being generated.

Later, as the metrics evaluator becomes familiar with and gains confidence in compiling information and producing reports using the metrics data, the role will focus more on how to enable the organization to make best use of metrics data. In the final stages of deployment, this person is likely to be working closely with a wide spectrum of managers to demonstrate how to make effective use of metrics data.

This role is often fulfilled by the project manager or project sponsor.

Table 3-1 profiles the types of people the metrics evaluator may want to contact.

<table>
<thead>
<tr>
<th>Job responsibility/title</th>
<th>Information these people are interested in</th>
</tr>
</thead>
</table>
| LOB managers, Department/Division/Group managers | ▶ What do the people that report to me need?  
▶ Who in my group is contributing/authoring?  
▶ What knowledge is relevant to my group/department?  
▶ What are “thought leaders” in my group reading, so I can have the other people in my group read those things? |
General skills which are assumed or required include that the metrics evaluator:

- Can analyze data to reveal trends, relationships, and patterns in that information
- Possesses business process knowledge (knows how data is used to manage the business)
- Possesses interdisciplinary knowledge (knows who does what in the company)
- Possesses overall knowledge of the types of data/data sources available
- Understands the value of information to the business their company is in
- Understands the company culture
- Can articulate (through information-sharing forums) how metrics can be used
- Possesses an affinity for wanting to help users do their job better
- Possesses competent (or advanced) computer skills

### K-map editor

This person could be a librarian or classification expert who is loaned to the team on a full-time or part-time basis – or simply an individual who is the most knowledgeable with regard to the existing data sources that will be leveraged by Discovery Server.

The K-map editor is responsible for overseeing the classification of information in the organization's K-map. This person is expected to provide specialist advice to the initial building and tuning of the K-map, and a reasonable level of technical competence is desirable.

Other duties for this role are:

- Selecting suitable data sources for building the organization's initial K-map

<table>
<thead>
<tr>
<th>Job responsibility/title</th>
<th>Information these people are interested in</th>
</tr>
</thead>
</table>
| Systems architects, systems managers, application managers | - Who is using which systems/repositories (Exchange publish folders, Notes DBs, and so forth)?  
  - Who is contributing?  
  - Are there problems with any of the systems?  
  - Do the right people have access to the data they need?  
  - Which databases are active versus inactive? |
| Chief Knowledge Officers, Chief Information Officers | - Are people in the organization sharing?  
  - What's the health of our sharing ability?  
  - What topics/documents are popular across the company?  
  - Who's accessing what kind of data? |
Labelling/re-labelling and creation of new categories
Validating the automated categorization of documents
Identifying relevant data sources for populating the K-map

One of the main challenges of this role is making sure the resultant K-map meets the business needs of the organization and is consistent with the scope of the project.

General skills which are assumed or required include that the K-map editor:
- Has a good understanding of classification/hierarchy/taxonomy concepts
- Possesses business process knowledge (knows how users interact with data and how it is used to manage the business)
- Possesses a sound knowledge of the type of data/data sources available
- Works comfortably with graphical user interfaces like Windows Explorer or equivalent, and understands conventions like drag and drop, cut and paste, renaming, and so on
- Possesses basic application development skills, to create field mappings and selection filters to help filter content appropriately for spidering by Discovery Server.
- Possesses sufficient technical competency to understand how the Lotus Discovery Server works (including concepts like affinities, clustering, and document scoring)

**Server administrator**
The server administrator is responsible for setting up and managing the infrastructure required for deploying the Lotus Discovery Server. Duties include installing and configuring the Lotus Discovery Server software, enabling the connections to the requested data sources, tuning and system maintenance, monitoring system performance, and providing technical advice. Effective execution of this role is crucial to the success of the Lotus Discovery Server deployment.

This person is likely to be someone who is an experienced Domino Administrator who also knows the larger organizational computing infrastructure as well. One of the main challenges of this role is dealing with the interplay of technical, organizational, and cultural issues that arise in a Discovery Server implementation.

General skills which are assumed or required include that the server administrator possess:
- Advanced knowledge of Domino administration
Advanced knowledge of Windows NT/2000 administration
- Detailed knowledge of the organization's IT infrastructure
- Knowledge of the organization's directory and authentication strategies
- Familiarity with major ERP and CRM systems/applications used in the organization
- Knowledge of IT security policy
- Knowledge of the organization's e-mail and document management policy
- Knowledge of the infrastructure used for collaboration in the organization (Lotus Notes, MS Xchange, Intranet)
- Knowledge of DB2 (or other RDBMS) administration functions; is (or has been) a database administrator (DBA)
- An understanding of taxonomy/classification concepts
- A sound knowledge of the types of data/data sources available
- Basic application development skills, to create field mappings and selection filters to help filter content appropriately for spidering by Discovery Server.
- Understanding of the value of information to the business their company is in

**End user**

Representation of end-user interests is usually accomplished through regular consultation with users, and through seeking feedback from users across the organization. In some cases, recruitment of users with specialist knowledge or skills may add a valuable dimension to the deployment team which could not otherwise be covered by existing team members. For example, promotional activities and education programs may well be best provided by users who have proven presentation and training skills.

**Tip:** A short-term assignment of selected users into the deployment team can provide valuable business perspective and context to team members.

At all stages of the Discovery Server deployment, end users should be encouraged to do the following:
- Provide feedback on usability and performance.
- Report problems on access, crashes, hangs, errors, and so forth.
- State expectations.
- Actively participate in pilots, trials, and so forth.
- Participate in, and contribute to, corporate programs and initiatives.
- Participate in establishing rollout schedules.
- Attend education programs and training sessions.

Note: The deployment team should ensure that mechanisms are put in place to facilitate user consultation and feedback.

3.4 Strategies for success

Building and sustaining a knowledge management solution can challenge even the most successful organization. As mentioned earlier in this chapter, it is important to identify a real-world problem, have the right personnel on your deployment team, carefully plan out your deployment, and have commitment from the organization as a whole. In addition to these strategies, this section discusses other key considerations that can help you achieve a successful implementation of the Lotus Discovery Server.

3.4.1 Read the documentation

We cannot emphasize enough the importance of reading the online documentation that is installed with the Lotus Discovery Server software. Long after this redbook has been published, the one source of information you can be sure is up-to-date is the online documentation shipped with the product. Of particular importance is the readme.chm file, which contains the latest release notes on features and issues to watch out for when installing and configuring the Lotus Discovery Server.

For a listing of all the key documentation sources that should be referenced, see “Product documentation” on page 297.

3.4.2 Obtain commitment

Experts agree that a knowledge management initiative stands little chance of success unless upper management takes ownership of the project. Project leaders can initiate the effort, but commitment from upper level management to get everyone behind it is an essential ingredient. As previously mentioned, the knowledge management project should also have a “visionary,” someone with credibility in the enterprise who can articulate the vision behind the initiative and set the strategic direction.
3.4.3 Do not ignore technology integration issues

As with any complex system, Discovery Server can touch many aspects of your computing infrastructure. It is important to plan for these technology integration and interoperability issues early on to ensure a successful rollout.

The infrastructure design and network impact aspects described in Chapter 4, “Infrastructure design and system performance” on page 79, as well as the user interface, directory/single-sign-on, and complimentary products aspects described in Chapter 8, “Integrating Discovery Server into your enterprise” on page 207, should all be carefully examined and considered during your planning stage.

3.4.4 Clearly establish the scope of the project

Establish the scope of any knowledge management project early, and try to be realistic in what you can achieve and in ensuring that it truly provides business value. Companies that deploy knowledge management systems hoping they will eventually discover a purpose for the system, or will realize unforeseen benefits after it is implemented, might wait a long time!

A Discovery Server solution, like any knowledge management effort, should be deployed to address a clearly defined business problem and to produce clearly defined benefits to your organization.

Section 3.1.2, “Finding an information problem” on page 60 discusses this aspect in more detail.

3.4.5 Encourage the right culture

It’s widely accepted that a favorable corporate culture is a prerequisite for the successful implementation of knowledge management initiatives, so you have to focus not just on technology, but on culture and process as well.

Sustainable knowledge management requires the introduction of a knowledge sharing culture organized around key skills, and a set of incentives to reuse knowledge to reinforce the skills required for the effective operation of the organization. You may not agree entirely with this point of view, but being aware of what is being written about knowledge management and considering how knowledge sharing and information reuse is currently practiced in your organization may reveal the type of culture you have to work with when you draw up your knowledge management program.
A key success factor in the implementation of Discovery Server is encouraging the right mindset in the staff that is responsible for delivering a knowledge management solution to the organization.

3.4.6 Involve, train, and inform your end users

The benefits of deploying a knowledge management solution are likely to be lost on most users in your organization because there is no immediate result you can point to. With the Discovery Server, you are rolling out new technology that requires a certain amount of education, end-user involvement in pilots, training, and promotion to ensure acceptance and effective use of the product.

In preparing your deployment plan, allocate time for the following tasks:
- Preparing users on privacy issues like spidering of mail files
- Informing users about how affinities and expertise location will operate
- Updating corporate policy to reflect changes in IT practices and information access

Users will be understandably apprehensive about the generation of knowledge and information that can be associated back to individuals. It is important to allay these concerns early, in the planning stages of deployment – rather than waiting until actual deployment and risking reactive hostility towards a new product that appears to be invading their privacy.

3.4.7 Obtain the right hardware

Enterprise commitment for any new technology initiatives can be severely tested if inadequate hardware resources have been allocated. Discovery Server does have significant resource demands, particularly in a large distributed implementation. System administrators should resist using second-hand hardware that does not meet the minimum system requirements.

Scale hardware requirements according to the solution being implemented. A high-end 700 Mhz desktop PC might be suitable for running a demonstration or proof-of-concept Discovery server, but reusing the same hardware for a multiuser pilot could jeopardize the future of your knowledge management program. Refer to 4.3 “Server hardware considerations” on page 93 for more information on sizing your hardware infrastructure.

3.4.8 Competing projects

Beware of deploying any major new technology initiative concurrently with other major programs the organization is working on. This is because IT personnel will
have their attention diverted away from your project, and the configuration, performance, and possibly the reliability of your organization’s IT infrastructure will be in a state of change. In addition, your ability to measure and monitor the performance characteristics of your Discovery Servers will be complicated by changes in the underlying infrastructure.

3.5 Summary

Overall, the key aspects of a successful deployment have been covered in this chapter. For anyone who has deployed new technology into an enterprise, the concepts of careful planning and phasing a deployment through test and pilot environments will probably not be new. Much of the content of this chapter should therefore be a quick refresher.

However, when knowledge management technologies are involved, it is important to always consider the cultural and organizational impacts. Finding a clear and defined purpose for your deployment, and then ensuring that the organization truly understands the need to solve that problem, is crucial to ensuring the organizational commitment to any cultural changes required.
Infrastructure design and system performance

This chapter discusses Discovery Server “server,” network, and application topology issues to help you design a Discovery Server computing infrastructure to match your needs.

It includes:

- Design considerations for network topologies
- Optimal distribution of Discovery Server tasks and services
- Security considerations
- Hardware, operating system, and Discovery Server specific performance tuning considerations
4.1 Discovery Server network design and distribution

In an ideal world, you would have an unlimited amount of server resources to allow every Discovery Server task to run on its own dedicated server; and all of your data repositories would be available on the same high-speed network segment. However, in most environments, the number of servers available for a Discovery Server deployment will be limited, there will occasionally be data sources to spider that are only available via slow WAN links, and other impediments such as locked down firewalls may be in the middle of your ideal Discovery Server infrastructure.

This section highlights some of the key considerations involved in deploying Discovery Server into such real-world environments, and provides a sample deployment across a fictional multiple-server infrastructure.

4.1.1 Primary and secondary servers

When discussing Discovery Server infrastructure considerations, it is important to clearly distinguish between the primary server and secondary servers.

The primary server is the server that:
- Controls all secondary servers
- Schedules tasks, consumers, and spiders
- Starts and stops all services and tasks on the secondary servers
- Holds the primary, writable DB2 replica of the K-map
- Maintains and monitors all queues (except the private ones which the spiders need when spidering a certain data repository), although every consumer uses a queue of its own on the server it runs on
- Is accessed to schedule and set all tasks, spiders, settings, and repositories
- Holds all Discovery Server databases to be replicated to the secondary servers
- Maintains the Discovery Server Control Center (discoveryadmin.nsf) database

The purpose of secondary server(s) is to:
- Run the K-map Indexing and K-map Building tasks
- Run spiders
- run profile and affinity services
- Act as K-map staging servers
While it is possible to run an entire Discovery Server infrastructure on one primary server, this is by no means recommended. Best practice Discovery Server environments have a primary server, and a pool of secondary servers across which to distribute the various Discovery Server tasks.

4.1.2 NSF replication between secondary and primary servers

It is important to note that secondary and primary servers do share some common NSF databases, and these databases must replicate regularly for proper secondary server function:

- The discoveryserver.nsf database should replicate one way from the primary to the secondaries with high frequency.
- The people.nsf database should replicate one way from the primary to the secondaries.
- The peopleq.nsf database should replicate bi-directionally between all servers.

This replication should be enabled by default. However, if odd behavior is experienced on secondary servers, such as no Discovery server tasks being loaded for an extended period of time, then the replication of these databases should be verified.

4.1.3 The impact of specific Discovery Server tasks

Even in the case of centralized servers within a single data center LAN, optimal distribution of Discovery Server services and tasks across your pool of primary and secondary Discovery Server servers is crucial to optimal performance. While it is very difficult to give exact rules or thresholds regarding such distribution of Discovery Server tasks to secondary servers, some general guidelines can be provided.

Table 4-1 is intended to give you a rule of thumb about how each Discovery Server service uses server resources. This list can then be used as a guide in determining your own environment’s distribution of Discovery Server tasks.

<table>
<thead>
<tr>
<th>Service</th>
<th>CPU</th>
<th>Network I/O</th>
<th>Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spiders</td>
<td>intensive</td>
<td>intensive</td>
<td>medium</td>
</tr>
<tr>
<td>K-map indexing</td>
<td>intensive</td>
<td>intensive</td>
<td>intensive</td>
</tr>
<tr>
<td>ATG</td>
<td>intensive</td>
<td>low</td>
<td>intensive</td>
</tr>
</tbody>
</table>
4.1.4 K-map building service (ATG) considerations

Tests have shown that the K-map building service runs faster on a single-CPU system than on a multi-CPU system. Therefore, we recommend that you let K-map-related services run on fast single-CPU systems. Choose processors with a high amount of first- and second-level cache over processors with high Mhz numbers, since the K-map building code can utilize cache memory to a large extent. Also use high-quality RAM modules with fast access cycles.

4.1.5 K-map replication considerations

Discovery Server 2.0 has introduced a new capability to replicate a read-only copy of the K-map to any number of secondary servers. Replicating the K-map can improve server performance by reducing end-user demand on the primary server's resources. However, it is important to note that K-map replicas on secondary servers still need connections to the primary server to communicate user interaction with documents that are metrics-related. In addition, K-map replicas on secondary servers are only updated every 60 minutes after the replica server has started with the default setting.

**Tip:** You're not prevented from enabling K-map replication during startup along with the other Discovery services, but it is recommended that you delay K-map replication enablement until you've built the K-map to reduce server workload.

4.1.6 Spider considerations

There are several key considerations when determining the optimal distribution of your multiple Discovery Server spiders:

- Discovery Server spiders should be distributed over several machines because all spiders use the same system resources (network and disk I/O). You should not create one secondary server to run all of your spidering.

- Initial spidering of a repository takes longer and consumes more resources than subsequent spider runs on already spidered repositories, since the subsequent spiders will only pick up new or changed documents. Do not be alarmed if the initial load of spiders on the servers to which you distribute them is higher than expected.
If you experience problems with repositories that take a long time to be spidered, the best thing to do is to think about adding another secondary server to move spider tasks onto if it appears to be a server resource issue. Spiders usually need high I/O performance capabilities, so upgrading any part of the server that relates to I/O performance will also help. For example, gigabit Ethernet adapters and high performance disk arrays are recommended for large-scale deployments.

Tests have shown that spiders perform better when running on multi-processor machines than on single processor machines. Increasing the memory (from 1 GByte to 4 GByte) did not have a remarkable impact on Spider performance. This applies especially to the file system spider and the Notes spider (all type of Notes databases including Domino.Doc and Quickplace).

Distributing a spider to an Discovery server on another network segment so that the spider can be closer to a given data repository will not have the expected effect. An example is the best way to demonstrate this issue:

- Consider a company with two intranets in two different countries (intranet A and intranet B).
- There are Notes databases running in both intranets (A and B), and initially the company deploys their Discovery Server servers only in intranet A, and only spidering data in intranet A.
- They have Notes databases in intranet B that they wish to spider. So that they do not have to spider over the WAN, they install an Discovery Server Notes spider in intranet B, while the primary Discovery Server servers stay in intranet A. They want the additional Notes spider in intranet B to process only this one Notes database in intranet B, which cannot be replicated to intranet A.
- However, you cannot tell which individual spider processes which data repository. So, the spiders in Intranet A will continue to occasionally spider the database in intranet B, while the new spider in intranet B will occasionally spider the databases back in intranet A.
- Deploying the remote spider in intranet B can actually increase the amount of spidering that occurs, as this new spider will actually spider data in intranet A.
- Overall, you cannot control which specific data repository a given spider will process. Any Notes spider can process any Notes database; any file system spider can spider any file system; and so forth.
4.1.7 Firewall considerations

With previous releases of the Discovery Server, there were occasions when firewalls would create an issue for an ideal deployment. In these earlier releases, all client access operated in a similar fashion to how the K-map editor operates in the 2.0 release (see 2.10.1, “K-map editor architecture” on page 53). That is, the users accessed the servers via client-based Java applets, which communicated directly back to the Discovery servers for the CORBA IIOP protocol.

Therefore in these past versions, if there was a firewall between your users and Discovery servers, there would often be a problem, since most firewall products do not provide support for the dynamic port requirements of the CORBA communications.

In release 2.0 of Discovery Server, this issue has partially gone away because the 2.0 architecture moved to a servlet-based approach for handling the end-user interface. User interfaces are handled via HTML/DHTML, and communicate to the UI servlets via HTTP and XML, as depicted in Figure 2-1 on page 16.

However, despite the architectural improvements in Discovery Server 2.0, firewalls must still be considered in the following circumstances:

- K-map editor access

  As described in 2.10.1, “K-map editor architecture” on page 53, the K-map editor access in release 2.0 still relies on direct CORBA (IIOP) access to the backend Discovery servers. Therefore, you will want to ensure that no firewalls exist between your servers and your K-map editor users. If this is not possible, then you should consider the CORBA firewall workarounds described in “CORBA over firewall options” on page 86 to provide your K-map editors with connectivity.

- Internet-based users

  Most deployments of Discovery Server are used for intranet/internal purposes. However, there may be occasions where Discovery Server end users will need or want to access Discovery services, such as search or K-map browsing, while traveling or at home via the internet. If this is the case, then the ideal situation would be to place the UI servlets in the DMZ, while all other Discovery services would be located behind the firewall on the company intranet, as depicted in Figure.
In this solution, extranet users access the Discovery Server servlets using TCP/IP through the firewall in the DMZ, while the servlets connect through the inner firewall to the core Discovery Server services using IIOP, via a point-to-point connection or other solution as described in the next section. The servlets would then be run on an application server in the DMZ – for example, WebSphere Application Server.

Finally, the core Discovery Server system is located in the intranet, along with its backend data repositories (DB2, file system), which do not necessarily need to be on a separate physical machine, but would usually reside on the Discovery Server’s machine. This illustrates that Discovery Server uses DB2, which could be put on a different physical machine, since Discovery Server’s file system is based on data stores, which can, of course, reside on a network file server.

However, this solution is not yet available with Discovery Server 2.0, as the UI servlets cannot be deployed to standalone servers as depicted, and the
backend data repositories cannot be located on separate systems. This option may be provided in future releases of Discovery Server.

In the interim, it is recommended that such internet-based clients connect back to the Discovery services running within the corporate intranet via an IPSEC VPN solution.

**CORBA over firewall options**

Standard firewall products are not well equipped for the protection of dynamic CORBA applications. They are tiered systems with a fixed configuration manually entered by the administrator. The access rules might state, for example, that hosts from the Internet are allowed to talk to the company’s HTTP and DB2 server. This works well for classic client/server applications because the servers always use fixed ports.

What makes CORBA different is that there is no fixed and well known port for CORBA applications and services. For simple applications, it is possible to bind a CORBA server to the same port in all instances. In this case the firewall can be configured to allow connects to this port. However, many applications and CORBA ORBs use server objects launched at random ports, as is the case with Discovery Server. To allow the execution of operations on such objects, the firewall must have open a large range of the upper IP ports.

One other firewall problem is in regards to CORBA application callbacks. The client executes an object and sends its IOR to the server (see “CORBA and IOR” on page 53). The server tries to connect to the object on the client side, but most client-side firewalls don't allow incoming connects, for very good reasons. For example, is the application call connect to port 6000 really an incoming callback or a hacker who wants to connect to X-server? The firewall has no chance to determine between friend or foe, so it is better to deny all such incoming connects.

A final CORBA firewall issue is the protocol for addressing the server. Normally, the IOR sent by the server to the client contains the hostname and the port of the server, but the firewall prevents the client from directly connecting to the server. The client instead has to connect to the firewall. This could be done by replacing the server socket in the IOR by the firewall socket, or by using an IOR that contains additional information about the firewall. In both scenarios this has to be done by the ORBs on the incoming side. Currently, many ORBs lack such firewall support.

To work around these issues, there are a few options for securing Corba connections:
1. **VPN IPSEC**

By utilizing an IPSEC VPN (virtual private network) to secure the connection between the CORBA machines, you can ensure that only the CORBA systems talk to each other through the firewall, and that the communications will be fully encrypted. This solution only requires that the appropriate IPSEC ports be opened on the firewall. However, IP SEC VPNs add additional cost, and can reduce performance and throughput.

2. **Open a large range of ports on the router**

This option will give you the best network performance of the three, but is the least secure. This option can be slightly secured by setting the routers to exclusively accept packets from the identified CORBA systems over this large range of open ports. However, this does leave you open to a potential attack from hackers that might “spoof” your CORBA servers.

3. **Implement a CORBA application firewall**

To solve the security issues around CORBA there are CORBA firewall products that proxy the transactions of the DMZ server to the firewalled host. This can be an effective way of securing CORBA communications through your firewall. The drawback is the configuration and test effort involved in implementing such a solution, and some loss in performance and throughput once such a solution is implemented.

### 4.1.8 Network traffic considerations

A Discovery Server infrastructure can generate a significant amount of network traffic during peak times of data repository spidering, indexing, and so forth. There are some key network aspects to consider when planning your infrastructure to minimize the impact of Discovery Server traffic:

- **Consider a dedicated, switched network.**

  Due to frequent IIOP communication between the Discovery Servers, you should try to put them in a dedicated, isolated subnet to maximize the speed of the communications. Additionally, it is recommended that you have a switched connection between the servers to ensure the traffic is isolated between the Discovery servers, and does not impact your general network traffic.

- **Minimize WAN links under 2 MBs.**

  If your data repositories change on a frequent basis, with a considerable number of changed and newly created documents, it is inadvisable to separate your Discovery Servers and connect them with slow links (2 MBit or less). It is also inadvisable to distribute a Discovery Server environment to distant geographical connections if the WAN connection to these locations is slow.
Give inter-server connection the priority over repository connections.

Spiders perform better when they are connected directly to the primary server via high-speed connections versus having a high-speed connection to the repository that is being spidered, and a slow link to the primary server.

Avoid network encryption if possible.

If you can afford to have reduced security on the inter-server communication, avoid tunneling and encryption between the servers. Encrypted data contains the extra overhead of the encryption, and will thus impact performance.

Be prepared for higher network utilization for new data repositories.

Your initial spider will create the most demand on your systems and network. Higher demand will also occur if there is a sudden change to, or addition of, a large quantity of documents. For example, if you are spidering a discussion board there will be frequent updates. This could be more demanding than a document library that has infrequent updates. After the K-map has loaded the documents, it then will be just spidering for updates.

4.1.9 A sample distributed Discovery Server architecture

Ideally, for large implementations, a dedicated server would run for each Discovery Server task. However, if you do not have this option, or believe it is not necessary for your implementation, then the following distribution of Discovery Server tasks and services across five servers is recommended:

- Primary (01) (ideally multi CPU)
  - Metrics collection
- Secondary (02) (ideally multi CPU, but single CPU would also be acceptable)
  - Profile source spider
  - Profile synchronization
  - Profile maintenance
- Secondary (03) (performs best on single CPU)
  - Metrics processing
  - K-map building
- Secondary (04) (ideally multi CPU)
  - Affinity processing
  - K-map indexing
  - XML spider
- Secondary (05) (ideally multi CPU, but single CPU would also be acceptable)
  - Notes spider
  - Web spider
  - Metrics reporting
Some important points about this example distribution of Discovery Server services:

- You do not need five servers to run a Discovery Server infrastructure. The example distribution is simply a recommendation of the most common distribution of services utilized in existing Discovery Server deployments. If a smaller numbers of servers is required, then combining the services on servers 02 and 03 and/or servers 04 and 05 would be recommended.

- Only metrics collection should be run on the primary server. This will ensure that the primary server is fully available for end-user access. If this primary server is overloaded with end-user access, then additional secondary servers should be deployed with K-map replicas to distribute the end-user load. Section 4.1.5, “K-map replication considerations” on page 82 discusses secondary K-map servers in more detail.

- Any spiders should be distributed over several machines because they all use the same system resources (network and disk I/O). In the distribution example provided, all possible spiders are not shown, but ideally spiders would be distributed over servers 02/03/04/05.

- On multi-processor machines, administrators should configure the number of spider threads to be a larger number than the default, perhaps on the order of 6 or so. This allows good utilization of the servers that are spidered instead of just running them two at a time. This takes advantage of latency in the network and the servers, and the distribution of content across multiple servers.

To help you better understand what a distributed Discovery Server architecture might look like when distributing services between separate, remote Intranets, see Figure 4-2 on page 90.
This diagram shows basically three things:

1. If you happen to have a Notes database (db.nsf) on a remote network (Intranet 2) that you want to have spidered by Discovery Server, we recommend that you create a replica of this database on a Domino Server on the same network as the primary Discovery Server (create replica “replica of db.nsf” on “Domino DB Server2” on “Intranet 1”) to reduce network traffic between the two Intranets.

2. To make the K-map browsing faster for users on the remote intranet (Intranet 2), place another secondary server with K-map replication (Discovery Server Secondary 4) on the remote intranet (Intranet 2). However, it is important to note that some level of traffic will still exist between the secondary K-map replica server and the primary server.

3. The secondary Discovery server hosting the FTIndex should be in the primary Discovery server's local intranet for performance reasons.

### 4.2 Discovery Server data and network security

There are obviously multiple aspects of security to consider when determining the security of any application infrastructure. We assume that you have already applied physical and operating system levels of security comparable to those
implemented in other parts of your infrastructure. Additionally, we assume that the existing data repositories which Discovery Server will spider are also properly secured, and that you are comfortable with how Discovery Server honors this repository security via the details provided in 2.5.11, “Repository security, and spider ACL collection” on page 35.

With these aspects covered, this leaves only topics specific to the security of data communicated between Discovery servers to discuss.

There are three areas to consider:
- Data security on the network in server-to-server transactions
- Data security on the network in client-to-server transactions
- Data security on the file system

**Data security in server-to-server transactions**

Server-to-server transactions themselves can't be encrypted in the 2.0 release of Discovery Server within the product itself. These transactions can carry document metadata susceptible to sniffing by experienced malicious hackers.

To ensure that all server-to-server transactions are less vulnerable to hackers, place all Discovery Servers together on a network segment that is not shared with users and non-Discovery Server machines. Make sure the router or bridge does not propagate server-to-server traffic off this segment.

However, transactions that carry work queue data can be encrypted. The work queues carry document content as well as metadata, so encrypting work queue transactions is an important aspect of Discovery Server security.

The key for work queue encryption is derived from the primary Discovery Server configuration document. This document is created when the first Discovery Server is installed, and is stored in Discoveryserver.nsf for all Discovery Servers to use.

To secure the work queue data, this server configuration document itself should be encrypted. This can be done most easily by encrypting the document with the public keys of one or more key administrators, especially yourself, and the public keys of all Discovery Servers.

See the *Discovery Server Control Center Guide (ds_admin.chm)* for details on setting up this work queue encryption.

In addition to using the Discovery server queue encryption capabilities to encrypt server-to-server traffic, the usage of IPSEC encryption of all network traffic between servers can also be considered for tighter security. IPSEC encryption of

However, full IPSEC encryption may have a performance impact, and such impacts should be carefully weighed against the need for increased security in your environment.

**Data security in client-to-server transactions**

Once server-to-server communications are secured as described in the previous section, it is important to consider whether client to server communications should be encrypted as well. By default, the K-map UI communicates over the HTTP protocol, so all client transactions are susceptible to sniffing by malicious hackers. In most deployments of Discovery Server, all users access the Discovery services via the corporate intranet, so such clear text access is often considered acceptable to corporate security managers.

However, should encryption of client-to-server communications be required, then the SSL protocol (i.e. HTTPS) should be enabled on the Discovery Server’s HTTP stacks to force all users to access the K-map UI via SSL. It is important to note that the implementation of SSL on any Web application can always have a significant performance impact, and this should be planned for accordingly.

**Data security on the file system**

To keep work queue contents from being observed, you can encrypt the data while in the work queues so that it’s protected when stored on a file system, as well as when it is transmitted over the LAN. Documents processed by e-mail spiders get encrypted automatically before being added to a work queue due to the confidential nature of their content.

However, encryption of work queues does carry overhead and may slow Discovery Server performance. If you find the performance cost of this too high, you can turn encryption off and use other measures, such as isolating Discovery Servers in a separate network segment, to protect your data.

**Important:** This procedure does not encrypt temporary files, so be sure to restrict access to the operating system’s Temp directories.

To enable work queue on-disk encryption, access the Notes.ini file for the primary Discovery Server in the Lotus/ds/ folder.

To enable spider work queue encryption, enter:

```
QueueCryptoLevel=high
```
4.3 Server hardware considerations

In addition to the distribution of the services, there are several considerations necessary regarding the server hardware to be used; they are discussed in the following sections.

Network interface cards
Since all the processing of data repositories and people repositories is done by communicating with the primary server, which is the only server in your distributed environment serving the master replica of the K-map (DB2), it is strongly recommended that you install high performance network adapters (preferably 100 Mbit or faster) in the servers. It is best to have 2 network interface cards (NICs) installed in a server, one for LAN connection and one for backup system connection.

RAM size
The size of the RAM in the Discovery Server is very important. Most Discovery Server services are highly memory intensive, especially metrics processing, K-map building and the spiders. We generally recommend RAM sizes of 512 MB up to 1024 MByte per CPU.

The primary server and the server running the K-map indexing task should have 4 GB of RAM, since the memory load on these machines is high. You could also start with 512 MByte per CPU and then measure the memory amount utilized by the tasks. If Windows starts caching large areas of memory to disk you should consider upgrading physical RAM.

CPU performance
For a productive Discovery Server environment, we do not recommend running any of the Discovery Server services on CPUs below 650 Mhz Pentium III Xeon level (or other CPUs with lower performance than that).

Furthermore, you should understand that no matter how many CPUs your server has, every service instance processes only one repository at a time. It is not possible to have two service instances process one repository concurrently.

Therefore, we offer the following rule of thumb for the number of services per CPU: For every CPU your servers have, you can have two instances of a particular service enabled (which is also the default value for every service).

Note: If you are running the Notes spider and the Web spider on a 4-way server, you can have, for example, 8 concurrent Notes spiders and 8 Web spiders enabled on that machine to spider data repositories.
Although the Discovery Server services automatically use all processors in an SMP server, you should always run performance logs to see how well the CPUs are used.

**Disk storage requirements**

You need to properly size the disk storage of your Discovery Server system – and this task depends on what your source data repositories are like, how many data repositories you think you will be using, and so on.

It is very important to have enough disk space available on the primary server because the primary server stores all queue files and DB2 databases. The queue files can grow large, especially if the data repositories that are spidered have frequent updates, and no worker task is working on the spidered data.

If you spider a large Web site and keep the consumer tasks (for example, metrics collection, K-map building, K-map indexing, and so forth) disabled, the queues will grow until there is no space left on the drives. The permanent data storage for the K-map is the embedded DB2 database.

It is not possible to estimate the exact size of the K-map and metrics in DB2, since this data depends on the size of the documents you spider, and all updates to those documents (document changes, metrics, responses, and so forth). However, the space needed for one document in your Metrics and K-map database is probably about an average of 30 Kilobytes, while some tests have shown that the required disk space needs to be 1.5 times larger than the original repository to spider over time.

**Tip:** A spreadsheet is included in the additional material for this redbook (see Appendix A, “Additional material” on page 295), which can provide a very rough order of magnitude estimate for the size of your Discovery Server tables. The disk size requirements for secondary servers are not as high as they are for the primary server, so they will differ from the values in this spreadsheet, which is geared towards primary servers.

To be able to dynamically extend available disk space, consider using consolidated disk storage for all servers (for example, IBM Shark arrays), where you install the executables and data stores of every Discovery Server, while installing the operating system to a local hard disk.

**Disk configuration**

Disk performance and disk failure are two issues that are important to Discovery Server. Since the down time of the primary server would impact clients even if they are on a secondary server, media failure is a critical issue. The other factor
you need to be concerned with is storage performance. Discovery Server is a very I/O intensive application; this makes high disk performance a requirement.

To protect against disk and disk controller failure, your choice is between mirroring and a disk array also known as RAID-1 or RAID-5. RAID-1 is the best choice for performance but is more costly per megabyte. RAID-5 is most common due to the cost savings when creating large volumes of data. A compromise would be to put the operating system on a mirrored volume and data on a RAID-5 volume.

The other issue is controller failure and distributing disk access across multiple buses in servers that have two system buses. This can cause loss of access to data. If you are looking for more performance out of your RAID system, consult your hardware vendor about implementing a tri-channel connecting your drives. In 8+ drive arrays this can boost you performance on RAID-5 configurations.

**Note:** Discovery Server is distributed with a run-time version of DB2. Due to the fixed implementation, splitting tables across disks or partitions is not available.

### 4.4 OS performance tuning considerations

Make sure your hardware is optimized for performance before considering this step; reconfiguring the OS is not the best way to correct for performance. In terms of tuning the operating system's performance, the following paragraphs contain general Windows NT and Windows 2000 tuning tips. Of course, all settings should only be done by trained system administrators and IT professionals. Please keep in mind that destroying your server's registry may lead to a major problem, requiring a re-installation of your operating system.

#### 4.4.1 Tuning Windows

Based on the assumption that your Windows server has more built-in memory than 128 MByte (which is strongly recommended), the following Windows registry settings should be taken into consideration:

1. "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Session Manager\MemoryManagement":
   a. "DisablePagingExecutive" = 1
   No parts of the Windows Kernel are swapped to the page file.
b. "IOPageLockLimit"
Set to a decimal value of “Ram size in kBytes divided by 8.” The default setting is 0. The optimal recommended setting (decimal) are as follows:

- 1024-2048 (16 MB)
- 2048-4096 (32 MB)
- 4096-8192 (64 MB)
- 8192-16384 (128 MB)
- and so forth

c. LargeSystemCache = 0
Reserves a smaller part of the page file for the system

2. “HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\PriorityControl\\Win32PrioritySeparation” should be set to “0”.

![Figure 4-3 Setting registry values for Memory Management via Windows registry editor](image)

### 4.4.2 Optimizing virtual memory

The pagefile is used to free physical memory, releasing content that is not currently needed and writing this content to disk. It is also used as virtual memory when your application experiences a shortage of physical RAM. This adversely impacts overall application performance, since the virtual memory resides on the machine’s hard disk, which is much slower than physical RAM.

The question of how big the optimal size of your pagefile should be is a very controversial question, but there are a few rules of thumb, which are described in the following paragraphs.
Set the minimum and the maximum size of the cache file to an equal value to avoid fragments in the file when new space has to be allocated. When your server has more than 512 MBytes of RAM, the size of your pagefile should be about 2 to 3 times the size of your server’s physical RAM (when running Discovery Server; for running other services, this value could be different).

![Virtual Memory](image)

Figure 4-4 Setting the pagefile properties (Windows 2000)

The identification of the location of the pagefile is a complicated decision and is specific to the number of hard drives. There are a number of ways to configure this, with each solution giving you a different level of performance. Table 4-2 presents some suggestions.

<table>
<thead>
<tr>
<th>Location</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optimal</td>
<td>Put a paging file on a hard disk (or multiple hard disks) that does not contain the NT system files or on a dedicated non RAID FAT partition.</td>
</tr>
<tr>
<td>Good</td>
<td>Put one paging file on a stripe set with no parity.</td>
</tr>
<tr>
<td>Adequate</td>
<td>Put one paging file on a stripe set with parity.</td>
</tr>
<tr>
<td>Worst</td>
<td>Put the paging file on the same drive as the NT system files.</td>
</tr>
</tbody>
</table>
Use the following steps set the pagefile properties:

1. Right-click My computer. Click the Performance tab.
2. Click Change. Click the appropriate drive.
3. Enter the initial and maximum size and click Set.
4. Repeat for all appropriate drives.
5. Reboot the PC.

File swapping performance is improved in relation to the number of physical hard drives used. Note that you must have multiple physical hard drives to take advantage of this feature. Simply spanning pagefile.sys across multiple, logical drives will have the opposite effect and will worsen performance on the system.

**Note:** Always try to have Discovery Server program files and data files installed on a physical hard disk drive other than the drive where you installed the operating system. It will improve your system's performance.

### 4.4.3 Tuning Windows

The following section is meant to help you increase the performance of your Windows operating system in general.

**Network**

As you might already know, every networking protocol installed on your machine causes the associated network search service to run, producing a considerable amount of traffic on your network. To optimize your operating system’s network performance, you should avoid installing unused network protocols.

The protocols should be listed by the frequency with which they are used. For example, the most used protocol should be first and the least used protocol should be last. Use the following steps to adjust the listing of network protocols:

1. Right-click Network Neighborhood.
2. Click Properties.
3. Click the Bindings tab.
5. Re-order the appropriate protocols.
7. Re-order the appropriate protocols.
8. Click **OK**.

Data transfer via TCP/IP can be increased significantly in networks with high bandwidths and long latency times by increasing the size of the packets. To achieve this, add the registry key "TcpWindowSize" (DWORD) to "HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Tcpip\Parameters". This value can be changed up to a maximum of 65535 Bytes; the default value depends on the NIC being used and is usually about 8000 Bytes. The size of the TCP windows controls the bytes that the system can retrieve before it sends out an acknowledgement. If you change this value, you should change it to an even multiple of the Maximum Packet Size (MSS). In an Ethernet network the TcpWindowSize value defaults to 8760 Bytes, which is 6 times the value of a 1460 Byte segment.

**CPU**

Windows NT's usage of the CPU's level 2 defaults to 256 KBytes. If your server's CPU has more built-in level 2 cache, you should increase this value to significantly improve your system's performance:

Registry Key: [HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Session Manager\Memory Management]

Value Name: SecondLevelDataCache

Data Type: REG_DWORD

Data: Cache (in Kb) using decimal notation

In Windows NT 4 you can watch the CPU load using the Task Manager, in Windows 2000 use the performance monitor in **Start -> Settings -> Control Panel**.
It is not sufficient to actually monitor whether the CPU is really overloaded or not. A CPU showing 100 % load is not necessarily overloaded. Much more important is the time that a thread has to wait before it gets to use the CPU.

To evaluate this value, you should use the system monitor.

4.4.4 Monitoring performance

To improve your Windows system’s performance you should consider monitoring certain system resources on a regular basis.

![Performance monitor startup screen](image)

Figure 4-6 Performance monitor startup screen

To monitor the CPU load, determine whether the CPU is overloaded or not, and to monitor if your server has enough physical memory and hard disk space available, just start the Performance monitor (Windows 2000) and right-click in the graph window area of the performance monitor window.

**Monitoring CPU load**

To monitor the load on your CPU and to be able to decide if the CPUs in your server are too slow, or if you need more CPUs in an SMP system, right-click in the graph area and choose Add.
In the dialog box, select the object “System” and the source “Processor Queue Length.”

If the average value is constantly above 4, this is a good sign that indicates that your CPU is not likely to be overloaded. The length of this queue is the most important factor if you have to identify a CPU that is too slow.

**Note:** The queue counter is only running correctly if you enable another indicator of the thread-object in the system monitor.
To see how much time of the overall CPU performance is used for non-idle tasks, the best indicator is the object “Processor” with the indicator “%ProcessorTime.”

Figure 4-9  Counter % Processor Time

If the biggest part of this graph is above 80 percent, a faster CPU is the required solution.

**Monitoring disk performance**

To be able to monitor the hard disk performance under Windows NT4 with the system monitor, this service has to be started first. Open a Windows command line and enter `diskperf -y`. A reboot of the system will be required.

**Note:** Only enable the disk performance monitor for the time of the actual performance test, since the disk monitor consumes a considerable amount of processor time. Disable the service with the command `diskperf -n`.

If your system constantly shows a high disk activity, this is likely to be due to insufficient physical RAM. Consider a RAM upgrade of your system. A good way to monitor whether your server’s physical RAM is too small is to watch the number of pages with the system monitor. The data source is the object **Memory**, where you should monitor “Page faults,” and “Available bytes.”
The first indicator tells you how many pages per second the system needs to perform to supply all threads with the required data. An average value of above 100 is an indicator of insufficient physical RAM. If 100 or even much higher values occur as peaks (few seconds to minutes), this is not an indicator of insufficient RAM.
An example of poor performance

Figure 4-12 is from a typical demo installation. All of the average values are pretty bad because the system is over the peak of its processing capabilities when put under stress:

![Performance Log File Sample](image)

**Figure 4-12  Load on a demo installation of Discovery Server**

To be able to find performance bottlenecks, you should consider establishing performance log files that are saved to disk and cover at least a few days of system performance.

If you are experiencing problems with a certain Discovery Server service, add this service to your log and monitor the load this service produces in the context of the already mentioned counters.
If you see a particular task consuming too much memory or requiring too much CPU power, you can better decide if you have to distribute this service to another server or if you have to upgrade your server.

In Figure 4-13 you see a very drastic example of what a performance graph looks like when you add the important processes of your Discovery Server system to the performance monitor.

**Note:** To be able to better distinguish between tasks on a crowded performance monitor screen, click the light bulb above the graph area and switch between counters, highlighting them in white color.

### 4.5 Discovery Server-specific tuning considerations

Within the *Discovery Server Control Center Guide* (ds_admin.chm), there is a detailed list of several specific Discovery Server performance-related settings that should be carefully considered for each large scale Discovery Server deployment. This list can be found in the ds_admin.chm under “Appendix A -> Troubleshooting -> Performance Tuning Tips”.

![Performance Monitor Graph](image-url)
Additionally, in 2.5, “Spiders” on page 20 there are some specific “performance considerations” listed for each type of Discovery Server spider.
Installation tips and troubleshooting

The installation process for the previous releases of Lotus Discovery Server caused a good amount of grief and confusion for those trying to install it for the first time. Missing or misinterpreting any of the steps and modifications involved resulted in frustrating attempts to complete the installation and setup. Launching the server console showed confusing error messages; some of them would disappear by simply restarting the server, while other errors had no other solution than reinstalling Discovery Server.

Fortunately, the installation process for Discovery Server 2.0 has been greatly improved. Extra pre-install checks have been included during the installation process to make sure all the prerequisites are completed, and that the right path to a successful installation is being followed. Additionally, error messages will challenge you every time the installation finds an offending condition. They contain clear information about what the problem is and the modifications you need to implement in order to continue the installation.

In this chapter we provide some practical tips on how to avoid problems during the installation of Discovery Server 2.0 via different installation scenarios, and how to troubleshoot the most common configuration problems related to Domino security and configuration of the Discovery Server in general.
This chapter doesn’t contain step-by-step guidance for the installation process; you can find detailed information in the product documentation. Hopefully, some of the tips found here will save you some trial and error experiences and will guide you to a successful installation.

Overall, this chapter contains the following:

- Tips and troubleshooting during the installation and setup in different installation scenarios
- Troubleshooting of the most common configuration issues
- Common configuration problems related to Domino security

### 5.1 Pre-installation considerations

Prior to performing any installs, all of the available product documentation should be read. The key documents that are installation-related are the install.chm and readme.chm files, which can be found within the root directory of the product CD or .zip file.

- The install.chm file contains detailed information on the system requirements, pre-install checks, modifications to the Domino, Sametime, and Discovery Server documents, and other configuration requirements.
- The readme.chm file contains the latest documented problems you can experience in particular scenarios and the workarounds for them.

#### 5.1.1 Prerequisites quick checklist

The rest of this chapter assumes you have read all the documentation and have met all the prerequisites specified in the install.chm file. However, to assist in ensuring you have met all these prerequisites, the following quick list is provided. Should you experience any problems during the installation, always come back first to this list and make sure you considered all the points. Before beginning the installation process, you need:

- A configured Domino server R.5.0.5 (or higher) to connect the Discovery Server to.
- A configured Sametime server (the certified versions are 2.0 or 2.5).
- A machine with the hardware and software required as specified in the Install.chm file. Notice the space requirements are 600 MB for the Program folder and 2 GB for the Data folder.
- The machine where you will install Discovery Server must not have Lotus Domino, Lotus QuickPlace, IBM DB2, or Microsoft IIS installed. It is recommended that a dedicated machine be used for Discovery Server.
A machine with Domino Administrator 5.05 or above installed must be available to perform changes to the Domino hub server (it can be in the same Domino server machine), the Sametime server, and the Discovery Server.

- A static IP address.
- A client browser with IE5.05 (minimum).
- A user account with Administrator rights to locally log on to the machine as describe in "Locally defined user account" on page 118. The password has to be less than 14 characters long.

**Important:** It is recommended that you create an image of your clean machine's operating system before starting to install Discovery Server 2.0, or that you back up the current Discovery Server or DS1.1 server if you decide to upgrade it. If anything goes wrong during the installation process, this will allow you to quickly reset the machine to its original state to try again.

## 5.2 Upgrading versus a new install

If you have deployed Lotus Discovery Server in your organization, based on previous versions of Discovery Server, you might consider an upgrade to Lotus Discovery Server 2.0. Here you will find a general description of the steps required for the upgrade from Lotus Discovery Server 1.1 to 2.0. For complete details on how to back up and upgrade from all the supported versions of Discovery Server, refer to the install.chm and read the readme.chm file.

Our recommendation though, is to seriously consider whether upgrading is what you need when looking forward in your deployment of Discovery Server in your production environment. If you have read Chapter 3 and Chapter 4, hopefully you have identified improvements that can be applied to solve problems in your organization and probably you are wondering how they can be implemented in your Discovery Server deployment. The easiest and most effective way is from scratch, after documenting all you have learned, what worked for you, and the problems you came across in order to find a solution for them this time.

Upgrading from previous versions is supported, but we encourage you to install Discovery Server 2.0 from scratch using the experience gained from POCs and pilots from using previous releases of Lotus Discovery Server, rather than upgrading them into your production environment. The effort involved in an upgrade many not be worth it.
5.2.1 Upgrading from Discovery Server 1.x

Upgrading from DS1.1 to DS2.0 requires the three manual steps described in the next section. Here we describe what needs to be considered and the order to follow.

- Read the readme.chm file for critical issues on upgrading from DS1.1.
- Since upgrading is irreversible, do a backup before starting the upgrade. Documentation describing how to manually back up data appears in the install.chm file and DS2.0 Control Center Help database.
- You will require 2 DB2 fixpacks for this upgrade install. Install the DB2 v7.1 FixPak3 and v7.2 FixPak5 in this order. Read the FixpackReadme.txt file before installing them.
- After they have been installed, you can start to install Lotus Discovery Server 2.0. You will notice the upgrade is very straightforward: many of the install screens are skipped, no DB2 is installed, and the Setup doesn't launch automatically because it automatically upgrades to your previous configuration settings.
- You need to upgrade to GTR 4.1 after the upgrade to DS2.0.
- Remember to remove the Temporary Internet files from your browser before launching the K-map. A complete explanation of how to do this can be found in “How to clean your browser's Temporary Internet files” on page 147.

Three manual steps to implement

Here we cover just the upgrade from DS1.1; for complete details on upgrading from other supported versions refer to the Install.chm file.

Back up your data

Before starting any of the supported upgrades, you need to remember they are irreversible, so a backup strategy is necessary in any case.

There are two types of backup available:

- Discovery Server data backup, which involves backing up data files and database tables, but no program files.
- Full mirror-image backup makes a full, disk-image copy of the contents of the DB2, DB2CTLCSV, and Lotus folders.

You need to decide which one suits your needs better. More information on backup and recovery can be found in the DS2.0 Control Center help, in the “Monitoring and Maintenance” chapter.
Upgrade the DB2 version
The two DB2 FixPacks required before upgrading to Discovery Server 2.0 are:

- v7.1 FixPak3
- v7.2 FixPak5 (to be installed in this order)

You can see what version of DB2 FixPack is installed on your machine by looking into the registry under HKEY_LOCAL_MACHINE\SOFTWARE, as shown in Figure 5-1.

This is just for informational purposes, but if you upgrade from a previous version to Discovery Server 2.0, you won’t be able to install Discovery Server 2.0 without installing these DB2 FixPacks first. If you ignore this step, you will get the error message shown in Figure 5-2 and you won’t be able to proceed with the installation.

Figure 5-1  Registry showing the current version of DB2 installed

Figure 5-2  DB2 FixPacks are required to resume the installation
They can be downloaded from:
http://www.ibm.com/cgi-bin/db2www/data/db2/udb/winos2unix/support/v7fphist.d2w/report#WINDOWS

For complete instructions on how to install the Fixpacks, read the FixpackReadme.txt file included in the DB2 Fixpacks.

**Attention:** Do not try to upgrade DB2 by running the setup.exe file in the ydb folder of the kit! This is not the correct upgrade.

Installation of the Fixpacks is very straightforward.

- They need to be extracted into a local folder on your machine.
- All DB2 services need to be stopped prior to installation.
- Install them by selecting **Start -> Settings -> Control Panel -> Add-Remove Programs -> Add New Programs**, then just provide the path to the folder where you extracted the FixPacks to and run from setup.exe.

The procedure to follow is the same for both Fixpacks. After installing them you can resume the installation of Discovery Server 2.0.

---

**Figure 5-3   Successful upgrade**

**Note:** If you have upgraded a distributed system and you are ready to start the servers, launch the primary server first and wait until it has completed the database design/refresh process before starting the secondary servers.
Remember to remove the Temporary Internet files from your browser before launching the K-map.

**Upgrade to GTR 4.1**

GTR 4.1 is the new version of the K-map Search and Indexing engine (the previous one was version 3.4). This new version provides the benefits of faster searches and better scalability to larger numbers of users.

**Note:** This step is done *after* you have installed the DB2 fixpacks and completed the upgrade to Discovery Server 2.0.

You need to remove your existing K-map index and re-spider all your content to create a full text index for the K-map to benefit from the new Indexing engine.

After you have upgraded to Discovery Server 2.0, use the following steps upgrade to GTR 4.1:

1. Shut down the server where the K-map Indexing service is enabled.
2. Back up the old index (ftdomain.di) tree, located in the data directory, then manually delete the ftdomain.di tree.
3. Edit the line in the notes.ini file to `FT_LibName=ftgtr40ds.dll`.
4. Restart the server.

   You will notice the following messages in the server console. This is normal behavior and a new index will be created after the upgrade.

   ![Server Console Output](image.png)

   **Figure 5-4** Messages returned during upgrade
5. Re-spider each repository so that all documents can be reprocessed at their next scheduled time by executing the following command at the primary server console:

   `load respider`

6. Pick a database at random in discoveryadmin.nsf; make sure that the “Process all documents during next run” option is checked. Then wait for the re-spidering to occur according to the set schedule to verify that re-spidering is functioning.

7. After all documents have been processed (which you can determine by viewing the logs), execute the following command at the server console:

   `load respider -c`

   This command causes all document values to be passed to the K-map Indexing queue for reprocessing the next time the Metrics Processing service runs.

8. Make sure the Metrics Processing service is enabled to run on one of the Discovery servers. The Servers & Services view, accessible from the Control Center's Maintenance navigator under System -> Servers & Services, shows which services are enabled on each Discovery Server.

9. Verify that Metrics Processing ran to completion by viewing the Metrics Processing log available from the list of Log links that appear along the top of the System Overview.

Afterwards, you won’t need any further configuration steps since the upgrading process will have made all the necessary modifications to the server documents and the Affinity Settings document to handle the new metrics processing configuration options.

**Note:** Any database indexes, such as people.nsf, will be reindexed automatically during the upgrade.
5.2.2 Upgrading from a beta build

Discovery Server 2.0 does not support upgrades from betas or test builds. If you want to install Discovery Server 2.0 on any machine that has a beta version of Discovery Server 2.0 or any previous test builds, it will be at your own risk. To avoid unwanted surprises, create a full backup. (Find more information about backup and recovery in the DS 2.0 Control Center help.)

Our advice, though, is that you fully uninstall the test build to install a clean version of Discovery Server 2.0 and create a new K-map, implementing all that you learned from using the beta build.

After uninstalling the previous version, make sure it has been removed completely.
5.3 Installing Discovery Server 2.0 for the first time

We assume that you have followed the information in the install.chm or Discovery Server Control Center stand-alone help files and that you have applied all the requirements specified there before installing Lotus Discovery Server 2.0.

Following *all* the steps in the documentation is the best warranty for a successful install and helps you avoid any problem during the configuration of Discovery Server 2.0.

**Attention:** Watch for misspellings and inconsistencies in your settings across servers, Hosts files, and so forth, during the multiple modifications you need to make. In many cases a simple typo can be the reason for you not being able to complete the setup of the server, or the cause of error messages you get, and you can waste your time wondering what the problem is.

Once you have installed and set up the first primary server, deploying secondary servers will be much easier because your Domino environment will be tuned up and the modifications to apply will be minimal.

**Tip:** It is advisable to create an image of your clean machine’s operating system before starting to install Discovery Server 2.0.

**Primary Discovery Server on an existing Domino domain**

The installation steps are very similar to the steps for other Lotus products, so we just comment on the ones that might cause a problem in the install, or when we can give a recommendation.

- Log on locally to the machine using a user account with administrative rights, as described in “Locally defined user account” on page 118. Afterwards, you will be able to change the user account to start the install.

  Install will issue a warning and abort if you logged in with a network account. The warning message is shown in Figure 5-6.
Launch the install from the setup.exe file on the Discovery Server 2.0 CD.

When you get prompted for where to install the Discovery Server 2.0 directories, we recommend that you install the Program folder and the Data folders LOTUS/DS/ and LOTUS/DS/DATA on different drives for better performance.

5.3.1 Specifying the user account for installation

This step verifies that you will use a user account that has the rights to complete a successful install. If the user account you use doesn’t comply with the
requirements, you will get an error message showing what the problem is and what the requirements are. Your choices are explained in the next paragraphs.

![Lotus Discovery Server 2.0 Installation](image)

**Figure 5-8 Specifying the user name**

- We recommend that you initially create a new, locally defined user account that is identical on all Discovery Servers you will install, to be used to install, load and run, and enable K-map replication. The account needs to be locally defined in the Local Administrators group with Administrative privileges and the Advanced user rights mentioned in the next section. If you decide to install using any other existing locally defined account, it will need the same Advanced user rights.

- Otherwise, you can create a new user account on the fly; it will be given the required advanced user rights automatically.

**Note:** In any case, the user account cannot have a space in the name.

**Locally defined user account**

If you decide to continue the install using a user account already defined on the machine, it needs to have Advanced User Rights allowing it to:

- Act as part of the Operating System
- Create a token object
- Increase quotas
- Replace a process-level token
These user rights are required by DB2 in order to create the CMAP tables.

- To check whether the locally defined user account has these advanced user rights, and to add them if missing, needs to be done from different locations depending on the OS to which you are installing Discovery Server.

On Windows NT:
- Go to Administrative Tools -> User Manager.
- Select Policies -> User Rights from the menu bar.
- Select the “Show Advanced User Rights” check box, then select the user rights you want to enable.

On Windows 2000:
- Go to Administrative Tools -> Local Security Policy.
- Expand Local Policies.
- Select User Rights Assignment.
- Select the user rights you want to enable.

- If you use the locally defined user account, enter the password associated with that account, do not enter a new password.
- The password you enter should not be longer than 14 characters, as NT requires.

New user account created on the fly
You can create a new user account on the fly, just by specifying a new user name not locally defined on the machine. The install process will automatically provide the required advanced user rights for a successful DB2 install and table creation.

- If you specify a new user name not locally defined, the required user rights will be added automatically. You create the password for it on the fly.
- The password you enter should not be longer than 14 characters

Note: Install does not support DBCS or accented user names.

After this step, the install process is very straightforward and should complete without any problems.

- The time required to set up the Discovery data store might vary depending on your machine’s processor speed. After this, you will see the progression of the files being copied across.
When the installation is complete and you have finished the product registration process, the setup process should launch automatically.

If this didn’t happen automatically at the end of the install, launch nserversetup.exe from the command prompt. Then, the browser launches and you can start to set up the Lotus Discovery Server.

1) It is required that this Discovery Server be part of an existing Domino domain. Have you registered this server in that domain?
   - Yes  - No
5.3.2 Setting up Discovery Server 2.0

This section contains a quick checklist of the steps required before starting the setup process:

- The Discovery Server must be part of a Domino domain.
- Make sure you have registered the Discovery Server on the domain’s Domino hub server. This is extremely important; the setup will fail if you proceed without having done this.
- Make sure you have made the changes to the Domino server, Discovery Server, and Sametime documents, as specified in the install.chm file.
- Make sure Discovery Server/Lotus Notes Companion Products is included in the Domino server’s NAB ACL with Editor Access and GroupCreator, GroupModifier roles.
- Have the Discovery Server ID handy; you will need to provide it during the setup process.
- If you are not sure what is the right format for your Server Name and Server Hostname, refer to “Common name = computer name = host name” on page 132.

Figure 5-12  Example of a primary Lotus Discovery Server setup
If you have followed all the steps in the documentation carefully, you should get the message that you have completed the Lotus Discovery Server setup.

After you click the Exit button, you should agree with the next two Internet Explorer messages.

![Internet Explorer message](image)

Figure 5-13   Click Yes to close the window

After rebooting your machine via the final install prompt, you can launch the Discovery Server console from the icon on your desktop or by selecting Start -> Programs -> Lotus Applications -> Lotus Discovery Server.

**Problems during the setup?**
Following are several common error messages, along with suggestions for discovering their cause and fixing the problems.

**Error message: Server not responding**
- Make sure you can ping the Domino server and that you can ping the Discovery Server from the Domino server machine. If you can’t successfully ping the host name, chances are the entry is not defined in your organization’s DNS tables. As an interim measure, you can create a local host table entry to resolve your host name to an IP address as explained in “Editing the Hosts file to resolve connectivity problems” on page 136.
- If you can’t ping it, find out what the problem is and correct it. Watch out for typos in your host’s files.
- Make sure you specified the right Name and Hostname for the server, as explained in 5.5.1, “Common name = computer name = host name” on page 132.
- If you need any assistance, use the Help buttons available during the entire setup process to access the DS 2.0 Control Center Help and hyperlinks to the readme file.
- Verify there are no misspellings in the server names, server documents, and IP addresses.
Error message: Can't find Server ID file. Please verify the server ID file exists in the location specified.

- Verify the ID file you obtained after registering the Discovery Server on the Domino server is in the path you specified in the setup document.

If for any reason you accidentally exit the Setup process before completion, try to launch \Lotus\DS\serversetup.exe from the command prompt. It might let you restart the setup process; otherwise, you will have to uninstall and reinstall.

**Note:** K-map Editor no longer installs with DS 2.0 since usually editing won’t be done from the server machine. Still, if required, it can be installed from the DS\DATA\DOMINO\HTML\ folder.

**Launching the Discovery Server**

The first time you launch the server console, you will see a list of messages. The server is checking for errors in the configuration and making any necessary modifications to run Discovery Server.

![Figure 5-14 Discovery Server checks and modifies the configuration settings](image-url)
Errors that need to be corrected

**Error message:** Server document: The Home URL has not been set to the proper value.

![Lotus Discovery Server 2.0](image1)

Figure 5-15 The Home URL has not been set to the proper value

Workaround: Go back to the Discovery Server document on the Domino server and modify the Home URL to the right setting. It must be set to:

```
/dsmain.nsf/fa_login?OpenForm&amp;/kmap.htm
```

**Error message:** Errors were found in the server document. Please correct these errors before running the Discovery Server.

![Lotus Discovery Server 2.0](image2)

Figure 5-16 Errors were found in the server document

Workaround: Go back to the Discovery Server document and check which is the offending condition in your settings. After correcting it, restart the Discovery Server and the message should have disappeared.

Other error messages that will need to be corrected are discussed in “People Awareness component (Sametime) not working” on page 137.

If you receive messages about enabling and loading SSO, see 8.2, “Directory integration and single sign-on” on page 213 for recommendations and troubleshooting.
Informational errors

The following error messages will occur the first time you launch your Discovery Server on the Discovery Server console. These errors are related to the fact that this is the first startup, and should not cause alarm.

**Error message:** Discovery Server: Unable to execute command (load dssched) on Domino server. Errors were found in the server document. Please correct these errors before running the Discovery Server.

Workaround: Restart the Discovery server console. It should disappear the next time you launch the server console if your settings are correct. Make sure you have added the group DiscoveryServers in the Discovery Server document in the Administrators field in the Basics tab.

**Error message:** Discovery Server: Unable to execute command (load dssched) on Domino server. You are not authorized to use the remote console on this server.

Workaround: No workaround is needed for this harmless error message. It will disappear the next time you restart the server console, as long as your settings are correct.

**Error message:** AMgr: Agent 'ConflictHandler' in 'lp\lpcat.nsf' does not have proper execution access, cannot be run.

Workaround: This message should disappear after you restart the Discovery server console. If it doesn’t, review the settings you input in the server documents against the documentation and correct the offending condition.

Figure 5-17 shows another harmless error message users might receive when accessing the K-map. Again, it can be ignored since the error message does not prevent users from accessing the K-map.

![Figure 5-17](image)

Fatal error message

If you get the error message shown in Figure 5-18 after installation is complete, and server setup is launching, you will have to unistall and reinstall the product. This error is related to an issue with improper setup of the Discovery Data Store (that is, DB2).
Use the steps described in 5.4, “How to uninstall Discovery Server” on page 130 to properly uninstall Discovery Server. Make sure you have deleted the folders DB2, DB2CTLSV, and CMUserTSP (CMUserTSP resides in the Data folder) as described in the uninstall steps, or the error may reoccur.

![Figure 5-18](image)

**Attention:** To shut down the Discovery Server console, the only command to use is `tell ds shutdown`. Do not use other Domino commands such as `quit`, `q`, `exit`, `restart server`, and so forth.

If after a while the server console hasn’t shut down completely, use the `Close X` option from the console menu.

### 5.3.3 Setting up K-map replication on the primary server

You won’t need to worry about enabling K-map replication until the K-map has been created and you want to make it available to your users, to balance the workload at least with one secondary server.

When enabling K-map replication, make sure you input a username with Administrative rights in both the primary server and secondary server machines.

![Figure 5-19](image)
Check the server console for possible error messages. If you see an error message like the one shown in Figure 5-20, use the following steps to solve the problem:

1. Disable the K-map replication and save the document.
2. Edit it again and enable it, this time using a username with advanced Administrator user rights in both the primary and secondary machines.
3. Save the document and monitor the server console. This time you should see messages that the replication setup has succeeded.

**Note:** Do not use a username containing DBCS or accented characters to set up replication between the primary and secondary server. DB2 doesn’t accept them.

Remember to do a full backup of your primary Discovery Server’s data store after the K-map replication setup has succeeded.
5.3.4 Secondary servers

Secondary servers are used for distributing the Discovery Server services and spiders within the Discovery Server environment, and for K-map replication to balance the workload of users accessing the K-map.

Once you have installed the primary Discovery Server, you shouldn’t have any problems with installing secondary servers since the installation requirements are the same. The setup just varies in that you point the secondary server to the primary server where the K-map will be replicated from.

Each secondary server needs to be registered on the Domino server and you need the ID for the setup.

![Lotus Discovery Server Setup](image)

1) It is required that this Discovery Server be part of an existing Domino domain. Have you registered this server in that domain?
   - Yes
   - No

2) Will this be a primary or a secondary Discovery Server?
   - Primary: The primary server manages and maintains the master copies of several Discovery Server databases, controlling updates to their replicas on any secondary servers. Only one primary server can exist per Discovery Server database group and it required.
   - Secondary: Secondary servers receive replica copies of Discovery Server databases from a primary server. Secondary servers, although optional, help to distribute and load balance the processing task.

3) Specify the name, hostname, location of the ID file, and password for this new server.
   - Server Name: ITSOLOD2/ITSOLOD2Domain2 (e.g., SrV1/KM/Acme)
   - Server Hostname: ITSOLOD2.bitus.com (e.g., SrV1.Acme.com)
   - Server ID:
     - Get from Address Book
     - Supplied in a file
   - Server ID: (e.g., itso1od2.id)
   - Server Password: *******
   - Confirm Password: *******

4) Specify the name of the primary Discovery Server to connect to.
   - Primary Server: ITSOLOD1/ITSOLOD1Domain2 (e.g., SrV1/KM/Acme)

![Figure 5-22 Example of secondary Discovery Server setup](image)

5.3.5 Setting K-map replication on the secondary servers

After you successfully set up K-map replication on the Primary Server, backed up the primary Discovery Server’s data store, and rebooted the primary and secondary server as specified previously, you can enable K-map replication on
the secondary server. The checkbox won’t be available if you haven’t enabled K-map replication on the primary server first.

**Note:** Before K-map replication is enabled on the secondary server, users still will be able to access the K-map if they point their browsers to the secondary server (they will be redirected to the K-map on the primary server). However, in order to benefit from workload balancing among servers, K-map replication needs to be enabled on the secondary servers.

- Select the K-map Replica checkbox in the Server document of each secondary Discovery Server. After you check this option and save the Server document, replication will copy the K-map data from the primary to the specified secondary servers.

```
06/18/2002 10:52:38 AM You are not authorized to perform this operation
[/discoveryadmin.nsf]
> load dbrep $ ITSOLD$1 >C:\DOCUME~1\ADMINI~1\LOCALS~1\Temp\83193695_TMP
> 06/18/2002 10:59:02 AM CMAP Replication: Cataloging the node : SRCNODE
> 06/18/2002 10:58:03 AM CMAP Replication: node SRCNODE has been created.
> 06/18/2002 10:58:03 AM CMAP Replication: Cataloging the database ...
> 06/18/2002 10:58:03 AM CMAP Replication: Database was cataloged.
> 06/18/2002 10:58:03 AM CMAP Replication: Source DGN was created.
> 06/18/2002 10:58:03 AM CMAP Replication: Password file was created.
> 06/18/2002 10:58:03 AM CMAP Replication: Creating control tables ...
> 06/18/2002 10:58:06 AM CMAP Replication: Control tables were created.
> 06/18/2002 10:58:07 AM CMAP Replication: Setting up the subscription ... 
> 06/18/2002 10:58:08 AM CMAP Replication: Subscription was set up successfully
> 06/18/2002 10:58:19 AM CMAP Replication: Initializing Apply, please wait...
> sq1code = 0 , sq1state = 000000
Apply initialization finished successfully.
> 06/18/2002 11:00:45 AM CMAP Replication: Continuing to set up the Apply ...
> 06/18/2002 11:00:45 AM CMAP Replication: Setting up the sleep time for Apply ...
> 06/18/2002 11:00:45 AM CMAP Replication: Sleep time was set up successfully.
> 06/18/2002 11:00:55 AM CMAP Replication: OpenSCManager succeeded.
> Starting... 
PASSED parm
Replication service installed
Ending... 
> 06/18/2002 11:03:56 AM CMAP Replication: Open Service succeeded.
> 06/18/2002 11:02:57 AM CMAP Replication: Service configuration was updated successfully.
> 06/18/2002 11:03:57 AM CMAP Replication: Setting up Apply status...
> 06/18/2002 11:04:08 AM CMAP Replication: The secondary server set up was successful.
```

Figure 5-23   Successful replication on the secondary Discovery Server
The primary server has to be running in order to access the K-map replicas on secondary servers

**Important:** When shutting down a distributed Discovery Server implementation, always bring down each of the secondary servers and then the primary server. When restarting them, start the primary server first.

### 5.4 How to uninstall Discovery Server

Follow these manual steps in any scenario where your run into trouble. These steps will uninstall Discovery Server 2.0, any previous version of Discovery Server, or any version upgraded to Discovery Server 2.0.

**Note:** If you are removing a version of, or upgrade from, Discovery Server 1.0/1.0a, the name of the Discovery Server folders in the path will be \KDS instead of \DS, so apply the uninstall steps accordingly.

1. From the Control panel, select **Add/Remove Programs**.
2. Remove Lotus Discovery Server.
3. You will get a message suggesting that you restart the machine to remove the files used during the uninstall. Select **No** since you need to manually delete other folders that haven’t been removed.

![Figure 5-24 Some of the folders left behind after the uninstall process](image)

4. Manually delete:
   - Lotus\DS and Lotus\DS\Data folders
   - DB2 folder
   - DB2CTLSV folder
   - DB2LOG folder created at un-install time

   The easiest way is to do a search on your machine for Lotus and for DB2 and delete the mentioned folders. Remember to reboot your machine after removing the folders.
5. Remove the Registry entry HKEY_LOCAL_MACHINE\SOFTWARE\Lotus\Raven if there have been previous test builds installed in the machine.

![Figure 5-25 Contents in the registry from previous installed builds](image)

### 5.4.1 What to do if unInstallShield fails

There is a known issue that sometimes occurs, in which the uninstall is unable to completely remove DB2. If your uninstall sheild fails, and it appears that DB2 is still installed, use the following steps to manually remove Discovery Server and DB2.

1. Stop the following DB2 windows services from the Services Control Panel:
   - DB2 - DB2
   - DB2 - DB2CTLSV
   - DB2 - DB2DAS00
   - DB2 - JDBC Applet Server
   - DB2 - License Server
   - DB2 - Security Server

2. Delete the Program and Data directories as explained previously. (That is, Lotus\DS and Lotus\DS\Data if you accepted the default selection during the install.)

   **Note:** Notice that if you are uninstalling LDS1.0 or 1.0a or after having upgraded from them, the folders still will be called KDS instead of DS, so apply the steps accordingly.

3. Delete the DB2 directories DB2 and DB2CTLSV.
4. Remove the DB2Log folder located on the c drive.

5. Remove the following registry keys from the Windows system registry if they exist:

   - HKEY_LOCAL_MACHINE\Software\Lotus\Raven
   - HKEY_LOCAL_MACHINE\Software\Lotus\Discovery Server
   - HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Uninstall\Discovery Server
   - HKEY_LOCAL_MACHINE\Software\Lotus\DS
   - HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Uninstall\DS

6. Restart the machine.

**Installation problems related to the DB2 version**

During installation, you may encounter an error message related to the DB2 version installed in your machine, for example:

"An incompatible version of DB2 was found. (Exit Install, remove this program, then resume install."

This occurs because the Windows Registry contains several entries that are not found with the initial search.

Once you remove the registry entries in the Windows Registry mentioned in "What to do if unInstallShield fails" on page 131, the Discovery Server installation program should work properly.

**Tip:** If, after following the previous manual steps, you still came across error messages during the install of DS2.0, the best option is restoring the backup image of the machine created before Discovery Server was installed. Use this tip in any install scenario when the problem is suspected to be failing to install because of a previous incomplete uninstall.

### 5.5 Top configuration problems

The following top configuration problems have been compiled from various sources, including quality engineers’ experience, consultants, customers reports, and so forth.

#### 5.5.1 Common name = computer name = host name

This naming issue causes one of the biggest confusions, especially when you are configuring Discovery Server for the first time. Use the following guidelines as a quick reference.
Server name
Its name in Domino hierarchical format, for example:
- Srv1/KM/Acme
- Srv4/Mil/M/Acme
- Srv10/Acme

The server name defaults to:
- <user's machine name> if this is a primary server
- <user's machine name>/<domain name from Primary Server field on previous page> if this is a secondary server

If you've cordoned off your Discovery Server implementation in a self-contained group within the main Domino domain, you'll probably need to adjust the name from the default values that initially display to reflect the special configuration.

Server's hostname
Its fully qualified DNS hostname format, for example:
- Srv1.Acme.com
- Srv4.Acme.gov
- Srv10.Acme.org

The default is based on the Windows NT settings.

This sounds simple enough, but this problem really caused users a lot of grief when setting up their Discovery Servers. A lot of the confusion comes from misinterpretation of terminology used in Domino and Windows.

First, let's define each of the three components.

Computer name
The computer name is what you called your Windows NT/2000 server when you set it up. It does not include the DNS suffix. In the example shown in Figure 5-26, the computer name is “ITSOLDS1”.

C:\>ipconfig /all
Windows 2000 IP Configuration
  Host Name : itsoldsi
  Primary DNS Suffix : lotus.com
  Node Type : Hybrid
  IP Routing Enabled: No
  WINS Proxy Enabled: No
  DNS Suffix Search List: lotus.com

Figure 5-26   Sample Windows 2000 IP configuration
Common name

The online installation documentation states you must register your DS Server name in your existing Domino domain using a name that exactly matches your WinNT machine name, but this is not quite true.

What you really need to do is register a Lotus Discovery Server with a “common name” that exactly matches your computer name (as defined previously). The key point here is common name. The safest and clearest way to describe common name is by example. Let’s start with a Lotus Discovery Server which has been registered with a hierarchical name of ITSOLDS1/ITSODSDomain2.

In this case:

- ITSOLDS1 is the common name.
- Our server has no organizational unit. (If it had one, it would be ITSOLDS1/OU/ITSODSDomain2, for instance.)
- ITSODSDomain2 is the organization name.

So, if your Windows NT/2000 computer name is ITSOLDS1, you could register any of the following Lotus Discovery Server names:

- ITSOLDS1/OU/ITSODSDomain2
- ITSOLDS1/ITSODSDomain2
- ITSOLDS1/xyz

The issue of what organizational unit (if any) and organization name you use is only important from a Domino server perspective. In a production environment you will probably register the Discovery Server in an existing Domino domain, and existing rules for organizational units and organization will apply.

In our demonstration environment, we have registered a server called ITSOLDS1/ITSODSDomain2.
**Host name**

The host name is defined here as the left-most part of the fully qualified Internet host name. Again this is best dealt with by example.

Let's take a fully qualified internet host name of ITSOLDS1.lotus.com. If we only take the first part of the name, up to the first ".", we get ITSOLDS1. This is the host name and it matches our Computer Name and our Lotus Discovery Server common name.

**Tip:** To verify the host name is responding okay, try pinging it from a command prompt window.

**Watch out for DNS**

DNS is another source of confusion for many administrators. The main issue to confirm before you set up your Lotus Discovery Servers is whether you can ping the host name from another computer. For example, Figure 5-28 shows the result of pinging a host name of ITSOLDS1.
Editing the Hosts file to resolve connectivity problems

Usually, adding the IP addresses of all the servers involved (Domino, Sametime and Discovery Servers), and including all versions of their hostnames in the Hosts files of each of your machines, resolves all connectivity problems among them - as shown in Figure 5-29.

Summary of connectivity

Table 5-1 summarizes how your Lotus Discovery Server name, computer name and host name work together. By now you should be able to understand why two of the examples are okay and why the other two are wrong.
Table 5-1  Common name = computer name = host name

<table>
<thead>
<tr>
<th>Discovery Server hierarchical name</th>
<th>Computer name</th>
<th>Host name*</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>ITSOLDS1/OU/ITSODSDomain2</td>
<td>ITSOLDS1</td>
<td>ITSOLDS1</td>
<td>okay</td>
</tr>
<tr>
<td>ITSO/OU/ITSODSDomain2</td>
<td>ITSOLDS1</td>
<td>ITSOLDS1</td>
<td>wrong</td>
</tr>
<tr>
<td>ITSOLDS1/ITSODSDomain2</td>
<td>ITSOLDS1</td>
<td>ITSOLDS1</td>
<td>okay</td>
</tr>
<tr>
<td>ITSOLDS1.lotus.com/ITSODSDomain2</td>
<td>ITSOLDS1</td>
<td>ITSOLDS1</td>
<td>wrong</td>
</tr>
</tbody>
</table>

* Fully qualified internet host name in this example is “ITSOLDS1.lotus.com”

Figure 5-30 shows where the names appear in the server documents.

5.5.2 People Awareness component (Sametime) not working

Figure 5-31  Example of People Awareness working
You won’t get the People Awareness feature to work in Discovery Server while the message in the Lotus Discovery Server console comes up:

11/07/2002 10:15:12 AM Errors found in Sametime configuration.

This means that there is something wrong in your configuration and you will need to go back over your settings and check for the cause. After you correct the problem, restart the server until the error disappears and you get the message highlighted in Figure 5-32.

<table>
<thead>
<tr>
<th>Time</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>10:15:12 AM</td>
<td>Verifying Sametime configuration.</td>
</tr>
<tr>
<td>10:15:12 AM</td>
<td>Errors found in Sametime configuration.</td>
</tr>
<tr>
<td>09:51:14 AM</td>
<td>Checking notes.ini for the correct Discovery Serv settings.</td>
</tr>
<tr>
<td>09:51:14 AM</td>
<td>Checking for an existing servlet directory.</td>
</tr>
<tr>
<td>09:51:14 AM</td>
<td>Verifying Sametime configuration.</td>
</tr>
<tr>
<td>09:51:14 AM</td>
<td>Sametime configuration check complete.</td>
</tr>
<tr>
<td>09:51:14 AM</td>
<td>Notes: If you’re setting up Discovery Server in a distributed environment, make a replica copy of the Secrets database (stauths.nsf) from the Sametime server onto every secondary Discovery Server. Create connection documents on the primary Discovery Server to push stauths.nsf from the primary Discovery Server to the secondary servers once a day.</td>
</tr>
<tr>
<td>09:51:19 AM</td>
<td>Single sign-on is not configured.</td>
</tr>
<tr>
<td>09:51:23 AM</td>
<td>JVM: Java Virtual Machine initialized.</td>
</tr>
<tr>
<td>09:51:23 AM</td>
<td>Java Servlet Manager initialized.</td>
</tr>
</tbody>
</table>

Figure 5-32  Successful Sametime configuration

Verify that you have changed the ACL in the Secrets database (stauths.nsf) on the Sametime server to include the group KDSAdministrators as a Person group with Manager access.

- Remember you will need to create a new replica of stauths.nsf onto the Discovery Server every time you install it. If you have reinstalled Discovery Server, a new replica needs to be created again.
- Verify the messages on the Discovery Server console about this database being successfully created.

Note: If you’re setting up Discovery Server in a distributed environment, make a replica copy of the Secrets database (stauths.nsf) from the Sametime server onto every secondary Discovery Server. Create connection documents on the primary Discovery Server to push stauths.nsf from the primary Discovery Server to the secondary servers once a day.

- Verify the connection document you have created on the Domino server to replicate (Push type) the Secrets database (stauths.nsf) from the Sametime server to the Discovery Server. The recommendation is to set it to replicate once a day.
Chapter 5. Installation tips and troubleshooting

139

Figure 5-33 Replication document from the Sametime server to the Discovery server

- In the Sametime Server document, be sure you have included the KDSAdministrators Group name in the “Run restricted LotusScript/Java agents” and “Run unrestricted LotusScript/Java agents” fields.

- In the Sametime server document Basics tab, select Yes in response to “Is this a Sametime server?”

Figure 5-34 Confirming a Sametime server

- Verify that you have added the Group Discovery Server/Lotus Notes Companion Products to the ACLs of the NAB in the Domino server, with Editor access and GroupModifier and GroupCreator roles.

- Every user involved should have the relevant Sametime server added in their Person document in the fully qualified Internet Host format, such as SameTimeserver.acme.com/DS or SameTimeserver.acme.org/KM. (This appears on the Basics tab on a Domino 6 server record; on the Administration tab in previous versions of Domino.) On our demonstration server we have ITSODSDomino2/ITSODSDomain2 because it has no OU and Sametime is installed on the Domino Server.
5.5.3 Cannot access the K-map editor

The K-map Editors group is created automatically when Discovery Server launches for the first time if it is not found in the NAB. Its contents are blank in this case.

Make sure the K-map Editor’s username is listed in the K-map settings document as in the following bitmap. K-map Editors can be added from the K-map Settings.
document (Select **System -> Groups** in the Maintenance navigator), or from the K-map Editors group in the Discovery Server’s directory.

![K-map Settings](image)

**Figure 5-37 Adding users to the K-map Editors group**

### 5.5.4 Replication problems and conflicts

Figure 5-38 illustrates the basic connection documents you should have between your servers in a basic Discovery Server environment.
We recommend you do all modifications to server documents and groups from the Domino Administrator client, making sure you select the NAB on the specified server you want to modify. Otherwise you will end up with replication and save conflict documents across the servers’ NABs.

If you get replication conflict documents, make sure you get the right information in the original document and delete the replication conflict.
Regularly check that replication between the servers occurs and there are no messages reporting the replication between servers failed. If you see error messages at your server console that indicate the replication failed, find out what the problem is and correct it, verifying the contents of the connection documents. Make sure, when you restart the server, the replication error message goes away.

To avoid replication conflicts, make sure you consider:

- You need to create a connection document (Push type) from the Domino server to the Sametime server (no need if Sametime runs in the same machine as the Domino server) to replicate the NAB to it.
- Notice you are required to create a connection document in the Sametime server’s NAB to push the Secrets database (stauths.nsf) to the Discovery Server once a day.
- For every secondary server you have in your environment, you need to create an extra connection to replicate stauths.nsf from the primary Discovery Server to the secondary Discovery Servers.
- The Discovery Server install automatically creates a Pull-Only connection document configured to pull updates from the Domino server to the Discovery Server every 60 minutes. (You can change this setting if needed to match your implementation preferences. Do this from the Replication tab in the connection document.) There might be a need to manually create this connection document in the event it doesn’t get created automatically.
- Replication errors for cross-domain certification environments occur if the Pull-Only connection document gets created but its Source domain and Destination domain fields on the Basics tab are left blank. If there is a replication issue for cross-domain environments, you should edit this connection document and fill in the Source and Destination domain fields with the correct information.

**Group names conflicts**

The same problem might happen with Group names, where you can get replication conflicts or Group names repeated several times. Apply the same solution: make sure the original document contains the right information and delete the duplication or replication conflict.
5.5.5 Issues with profiles

Users might report problems when trying to access their own profile or other people profiles from the K-map.

Problem: The following message is received by a K-map end user:

Profile not found for <name>

The problem might be caused by copying and pasting large amounts of text, especially HTML text, into a field of the person’s profile. Doing so might trigger this message when attempting to display another section of the profile. It might cause Java Script error messages as well, so this is to be avoided.

Problem: The following message is received when a user tries to display profiles:

No profile was found. Message HTTP Web Server: Lotus Notes Exception - Entry Not Found in Index

The profile source directory (either Domino or LDAP) may contain Person records with no first names (last names only). Update the source with first names to correct the problem.

5.5.6 Problems reported in the Affinity Processing log

You need to have enabled Affinity Generation first, then information will display in the Affinity Processing log after the affinity processing service has run.

The Affinity Processing log is a good source of information to identify problems with affinity generation and e-mail notification to the users, so we recommend that you check it on daily basis to see that e-mails are sent, profiles are updated, and to review any warnings that may appear.
Problem: E-mail address is blank, thus the user won’t get the affinity proposal mail notification.

Problem: E-mail address is blank, thus the user won’t get the affinity proposal mail notification.

5.5.7 Spidering file systems

Trying to spider file systems has caused problems because you need to have the right access to the file system in order to spider it successfully.
If you want to spider file systems on NT and Windows 2000, the machine's drive or folder has to be shared, and you need to have Read permission to map it from the machine you want to spider from.

![Filesysshare Properties](image)

**Figure 5-42  Sharing a folder to allow file system spidering**

From the Discovery Server machine, map the machine to the drive or folder that contains the files you want to spider. You should see the machine, the shared drive, and its contents when you browse from the Directory selector, as shown in Figure 5-43.
If the spidering results in a Fatal Error message, check the log to see what the problem is. There are different access rights requirements depending on the file system types. If you have an access problem, you need to find out what access rights you need and request them from the relevant system administrator in order to allow you to spider that file system.

- FAT and FAT32 file systems do not have directory- and file-level ACLs. The only access information used is the share access lists.
- To collect ACL information about NTFS file systems, the spider system must have administrative access to the system being spidered.

### 5.5.8 Browser-related issues

**How to clean your browser’s Temporary Internet files**

Internet Explorer stores visited pages in a cache so that they can be retrieved quickly if you view them again. This feature can be misleading sometimes: if you get a problem on the browser that you can relate to some wrong settings in your Discovery Server configuration, you will get the same problem again even after the settings have been corrected. The reason behind this can be that the browser is loading the same page again, trying to improve its performance.

There is an easy way of changing the browser’s settings to make sure the information you get is the most up-to-date. Use the following steps to do this.
1. From the Internet Explorer menu bar, choose **Tools** and then **Internet Options**.

2. In the Temporary Internet Files section, click **Settings**.

3. Specify **Automatically** as the interval to check for newer versions of stored pages.

4. Click **OK** to close each window, first the Settings window and then the Internet Options window.

5. Close and restart the browser.

Moreover, you will need to clean up any Internet Explorer browser clients used to access the Lotus Discovery K-map before accessing the K-map after a new installation. Using Windows Explorer:

1. Locate the Downloaded Program Files directory in the Windows (\Windows) or Windows NT (\WINNT) directory.

2. Select the K-map, LncchCtIClass, and KmapCtIClass files, right-click, and select **Remove** (*not* Delete!).

3. Empty the client's Temporary Internet files folder.

**Security-related issues**

The browser's security level settings can affect the cookie storage for K-map functionality in many different ways if they are set too high.
The remedy for this is to reset the IE security settings, using the following steps, after logging into Discovery Server from the browser:

1. Check the browser's status bar to determine in what Web content zone the Discovery Server is located (Internet or Local Intranet).

2. From the browser menu bar, choose Tools and Internet Options.
3. Click the Security tab to display the browser's security settings.
   The default Web content zone for the Discovery Server is selected.
4. Set the security level for the selected zone to Medium.
5. Click OK to save the new security settings, then close and reopen the browser.

Note: If users still experience problems, they might need to change the default Web content zone. Make sure that Internet zone is selected if the URL for your Discovery Server contains periods. Browsers might be set up to allow certain Internet sites to be a part of the Local intranet zone. If this is the case, choose the Local intranet zone for the security settings and experiment with security settings in both Web content zones.

5.5.9 KDSAdministrators group is missing important entries

The KDSAdministrators group is an important group that is used in database ACLs and in several places in the Server Document to allow privileged access to all parts of the Lotus Discovery Server.

This group is created if missing, and is populated the first time Discovery Server is launched with the required users for Discovery Server to run. Figure 5-46 is an
example of the contents of the KDSAdministrators group that gets created automatically.

![Figure 5-46 Example KDSAdministrators Group](image)

Notice, though, that at least an administrator person needs to be added manually to this group in order to perform the administrative tasks from the Lotus Discovery Server Control Center!

Access to the Lotus Discovery Server Control Center from the browser is the same as it was in the previous release DS1.1; see an example of our server at:

http://itsolds1.lotus.com/discoveryadmin.nsf

At minimum, the KDSAdministrators group should contain:

- The primary Lotus Discovery Server name, for example:
  ITS0LDS1/ITSODSDomain2
- The “Discovery Server/Lotus Notes Companion Products” entry
- At least one individual hierarchical user name, for example:
  ITS0DS0mino2 Admin/ITS0Doma1n2
- Add to this group just users that should have access to the Discovery Center Control Center.
Chapter 5. Installation tips and troubleshooting

5.5.10 Cannot access the K-map

This is a new feature in Discovery Server 2.0, rather than a problem. (However, if you actually get the message shown in the example screen, it would be a real access problem.) In previous versions, the K-map was accessible and searchable immediately after being created by ATG, what was not very impressive to users due to their expectations about the K-map results. The raw results were available during the entire editing process, with users able to watch all the relabeling and movement of categories, which could be even more confusing.

In this release the K-map isn’t be available “for public viewing” until the K-map Editors are happy with the K-map or parts of the K-map and decide to publish the whole thing, or part of the categories tree.

Important: The Add Groups dialog box in the Maintenance tab that appears in the Lotus Discovery Server Control Center doesn't prevent an administrator from accidentally removing all members of the KDSAdministrators group. Make sure that at least one administrator is listed in this group.

Figure 5-47  System Groups maintenance screen

If all members of the group were accidentally removed, open the server's copy of the domain address book and add the default members back to the KDSAdministrators group.
Until this stage is reached, the Search feature will be available to users as long as the initial K-map has been generated and indexed.

**Figure 5-48  K-map not available message**

### 5.6 Common configuration problems related to Domino security

#### 5.6.1 Access to the Domino Directory

This is a security issue pertaining to who is allowed to modify the ACL of the Lotus Domino Directory in the domain where the Lotus Discovery Server has been registered. During installation of the Lotus Discovery Server, the installation program attempts to automatically create, or if they already exist, edit several groups in the Domino Directory. This is only possible if you have added the entry "Discovery Server/Lotus Notes Companion Products" to the ACL with Editor access (with group creator/modifier roles) to create groups and update existing groups.

Talk to your IT security personnel early on to ensure access to the Domino Directory does not become an issue during installation and setup of the Lotus Discovery Server.

To avoid this problem, get your Domino administrator to create the groups identified in Table 5-2 in the Domino Directory before installing your first Lotus Discovery Server in the Domino domain. Next, populate each of these groups with the appropriate members.
### Table 5-2: Special groups created in the Domino Directory

<table>
<thead>
<tr>
<th>Group name</th>
<th>Sample members</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>KDSAdministrators</td>
<td>ITSODSAdmin2/ITSODSDomain2</td>
<td></td>
</tr>
<tr>
<td>(Individuals authorized to initially configure and maintain all DS services)</td>
<td>Discovery Server/Lotus Notes Companion Products</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ITSOLDS1/ITSODSDomain2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ITSOLDS2/ITSODSDomain2</td>
<td></td>
</tr>
<tr>
<td>AffinityDesignators</td>
<td>User12 Name12/ITSODSDomain2</td>
<td></td>
</tr>
<tr>
<td>(Usually supervisors or managers who are knowledgeable about the Category content in the K-map. Empowered to recommend affinity associations with specific categories in the K-map)</td>
<td>KDSAdministrators</td>
<td></td>
</tr>
<tr>
<td>DiscoveryServers</td>
<td>ITSOLDS1/ITSODSDomain2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ITSOLDS2/ITSODSDomain2</td>
<td></td>
</tr>
<tr>
<td>KmapEditors</td>
<td>User13 Name13/ITSODSDomain2</td>
<td></td>
</tr>
<tr>
<td>(People who have subject-matter expertise and can participate constructively in editing and relabeling categories in the K-map).</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Tip:** Only specific people should have access to the K-map Editor application as members of the KmapEditors group. Since all K-map editors can see all document information, this access needs to be protected. We suggest you don't include the KDSAdministrators group in the KmapEditors group. If an individual administrator needs to be able to participate in the K-map editing process, then add the administrator's individual ID explicitly to the group.

### 5.6.2 Server security settings are incorrectly set

The server document contains a number of security settings which control access to Java applets, servlets, LotusScript, and HTTP browsing. Refer to the install.chm file or DS 2.0 Control Center Help, where you will find a detailed explanation of how to set up your security settings in the server document. If you get this right, you will eliminate a lot of commonly reported errors.
5.6.3 Cross-certification problems

Spidering of databases and directories in foreign domains

If you want to have access from Discovery Server to spider Notes databases or a Domino directory in a different Domino domain, first you will need to create cross-certification between the two Domino domains.

Basically, you need to cross-certify IDs in order to get your domain certified in the foreign domain and the foreign domain certified in your domain.

In addition, you need to create a connection document between the Domino server in your domain and the Domino server in the foreign domain.

This is explained in the Domino 5 Administration Help file.

Figure 5-50  Error message regarding cross-certificates
To spider Notes repositories or Domino Directories in a foreign Domino domain you must manually type the name of the server and database you want to spider because the foreign Domino server won't appear in the selection list.

If a user tries to open a Notes document from the K-map and receives the message "Server Error: The server's Address Book does not contain any cross certificates capable of authenticating you," this means the repository where the document resides is on a server where the user is not cross-certified.

To access documents in a repository in a foreign domain, users must have their IDs cross-certified with the repository's server.

**How authentication of users works**

This is an overview of how user authentication works in Lotus Discovery Server.

All types of users need to supply a user name and password to authenticate with Discovery Server. The destination logged into is determined by the type of user:

- End-users log into the K-map.
- KDS Administrators log into the Discovery Control Center.
- K-map Editors log into the K-map Editor.
The user account is checked by standard Domino methodology, using the Internet password contained in the user's Person document. The user needs to be listed in the primary Domino domain or in a Foreign Domino domain accessible by Directory Assistance to get the authentication information from the external domain.

Group expansion (which means getting a list of groups a user belongs to) occurs in the following way:

- In the primary domain it is handled by Domino and occurs when the user authenticates.
- For Foreign domains, group expansion isn't handled by Directory Assistance. Discovery Server performs its own group expansion if cross-domain access checking is enabled as explained in the next section.

**How to enable cross-domain access checking**

To support cross-domain access checking, the Discovery Server needs to know the name of the server that has the Domino Directory for group expansion in the external domain.

This is done by the administrator of Discovery Server with the following steps:

1. Create a local replica of the foreign server's Domino Directory, so that the system can still function if the external server is unreachable for any reason.
2. List this [replica] server in a Notes.ini variable as follows (one line for each foreign domain). The integer suffix is required and should be unique for each such entry.

   ```
   ExternalDominoDomain1=<Domain name>,<Server name>, <optional directory filename.nsf>
   ```

   For example:

   ```
   ExternalDominoDomain1=Acme, ServerA/Acme
   ExternalDominoDomain2= Acme, LocalServerB/MyDomain, aceNAB.nsf
   ```

At user-authentication time, Discovery Server accesses this server for the information and creates the group lists.

**5.7 Additional information**

If you are still having problems, see "Related publications" on page 297 for other sources of troubleshooting information and assistance.
Developing and maintaining an enterprise taxonomy

Whether your organization intends to take advantage of the full set of Lotus Discovery Server features or just use its search functionality, you will need to work with your taxonomy. The Planning chapter oriented you to some important concepts. In this chapter we take you from understanding what a taxonomy is from a technical and historical perspective to giving you all the tips and techniques you need to build a robust, enterprise-strength taxonomy including:

- Taxonomy definitions and history
- Planning, creating, and editing a taxonomy
- Choosing repositories
- K-Map Builder technology and settings
- Field mapping
- K-map Editor - detailed techniques for editing your taxonomy
- Categorization
- New tools to maintain and enhance your taxonomy
6.1 Taxonomy definition and history

In her book *Practical Knowledge Management*, Wendi Pohs, resident “cybrarian” and taxonomy expert on the Lotus Discovery Server development team, gives an historical perspective of taxonomies: “...organizations use taxonomies to categorize information and improve communication. Historically, biologists have been most familiar with taxonomies. The International Classification of Diseases (ICD) has existed since the 19th century and is still used and analyzed today. In 1956, a committee of educators published Taxonomy of Educational Objectives to improve and standardize their instructional techniques. Currently, a good example of a taxonomy is the Yahoo! Web site.”

Traditionally, taxonomies have been structured manually in advance, for example the way a library classifies its books on shelves. A rule-based classification system such as this is easy to understand but is sometimes limited when categories have to be added to the existing scheme. For example, consider the rule-based diagram of race distances in Figure 6-1:

![Figure 6-1 A rule-based diagram of race distances - 1](image)

The second level of this classification scheme has categories of race distances based on distance in miles. But what about recently popularized metric-based race distances such as a 10 kilometer race? We could add a category for a 6.2 mile race but wouldn’t it be better if we could add a category labeled 10 Kilometer or “10K”, which is its popular definition? Also, consider your users. In this example, high school runners don’t compete at the longer distances that college and Olympic runners do. How about a more informal, or natural, classification based on the needs of the runners, as follows (see Figure 6-2 on page 159):
Figure 6-2 is more representative of the shorter distances for high schoolers while also allowing for conversion to metric distances as needed.

The automatic document clustering engine of Discovery Server will create a taxonomy for your organization using natural classification schemes. Your human taxonomy editors can modify these schemes to meet the needs of the people in your organization.

The people who edit your taxonomy, and the Lotus Discovery Server with its Automatic Taxonomy Generator (ATG), are where art meets science. Actually it’s where the human mind meets technology—by working together, a powerful Knowledge Map (K-map) can be created for your organization. The K-map is a visual representation of your content, organized and structured in a way that complements full text search, helps your users navigate to the precise information they need, and creates affinities between user profile and document information. While Chapter 2, “The Discovery Server architecture and key components” gave you a technical analysis of the ATG, the remainder of this chapter guides you through creating and working with your K-map.
6.2 Planning, editing, and maintaining a taxonomy

The logical reason for planning, editing, and maintaining your taxonomy in a sensible fashion is to maximize the investment you’ve made in the Lotus Discovery Server. By closely following the recommendations in this chapter you will have a K-map that is generated and, more importantly, efficiently updated by the K-map Builder service of Discovery Server. (The terms K-map Builder and Automatic Taxonomy Generator (ATG) are synonymous.) In other words, let the ATG automatically create your categories, while you apply the finishing touches like an artist to create the overall hierarchy. K-map Builder is the term you’ll see in the Discovery Server marketing material and product documentation. ATG is the actual Discovery Server program/task that creates and helps you maintain your K-map.

It's certainly okay to do as much manual editing as you want to your K-map but keep in mind that the ATG’s algorithms may not understand your human logic if you scramble up the initial K-map too much by moving individual documents all over the place. “Retraining” a category is a powerful new feature with Discovery Server 2.0 that helps alleviate this scenario, but as you read on you'll realize there are ways to make the ATG work for you rather than to be a slave to the technology yourself.

In the Planning chapter you were given a standard recommendation for members of your taxonomy team. If your company has a Knowledge Management group, a Chief Knowledge Officer, and a taxonomist or two, great! If not, you do not need to create the functions and hire people. A team of people who understand your organization and its electronic content will suffice.

For a full Discovery Server implementation it is imperative to create an effective K-map to generate accurate affinities so your organization can utilize its expertise (affinity generation and expertise profiles are handled in depth in the Affinities and Metrics chapter). A refined K-map also allows users to efficiently navigate it for knowledge, expertise, and awareness using the K-map UI.

6.3 Complete steps to fully utilize a taxonomy

Now that you have some background on taxonomies and understand the importance of leveraging the power of Discovery Server correctly, the sections that follow offer detailed tips and techniques for getting underway with your taxonomy:

- Choosing repositories
- K-map Builder
- Field Mapping
Choosing repositories

Choosing repositories to include in your K-map is an important, but not overwhelming, step that requires some planning. Easing your work is the fact that Discovery Server handles many different types of repositories that are detailed in depth in this section. The spiders, described in the Architecture chapter, can even read attachments using Key View Filter technology. The ATG will then categorize them appropriately. You have the option of letting users open the container document for the attachment or the attachment itself.

Please note that zip files are an exception. In the case of a zip file attachment, hopefully a container document such as Notes or Domino.Doc includes some text within its body that describes the content of the zip file. Then Discovery Server will do its work and categorize the content correctly. When users access the document, they can extract the zip file if they desire.

Important: If your organization is planning to utilize the expertise feature of Discovery Server, then it is important to note that how user names were originally defined and maintained in your repositories can create additional work. Give some thought to generating user profile documents from a directory source that contains all of an individual's aliases. It is certainly possible to add aliases later in your DS deployment, but it will require additional work for your administrative staff who will have to edit the alias field in each user's profile form since an end user does not have access to do so.

By following the recommendation in the Planning chapter to use a “training set” of 3000 to 8000 documents (but never more that 25,000 documents or you’ll run out of memory), you’ll begin to see how Discovery Server handles different types of content. Here’s a list of repositories ranked according to the strength of their attributes for spidering with Discovery Server:

1. Generic Lotus Notes databases
2. Lotus Domino.Doc databases
3. Lotus QuickPlaces
4. Windows-compatible file system files
5. Web sites
6. Two new spiders in Discovery Server 2.0: the XML and MS Exchange spiders. These spiders are listed last in the list not due to a limit in their capabilities, but due to the fact that they are new capabilities, and thus less known and tested.

From the start Notes databases have always had good metadata. For example, Notes has always offered large body fields for rich text; the view feature encouraged people to include title and category fields; and even in the rare instance of an application designer neglecting to include an Author or Owner field on a form, Notes keeps track of people who have updated a document in the $UpdatedBy field. Some Notes applications have forms that are loaded with hidden fields to control the behavior of that particular form (as long as the hidden field is not a computed one and stores its value in the document, you can spider it). By working with the application designer of a database, you can uncover great metadata to use. Also, if your organization wants to spider e-mail, the Memo form always has a required From field.

An interesting thing to note is that .nsf attachments are included in the default exclusion list for the Notes spider. Logically, the Notes spider will not process an .nsf within an .nsf. Again, as with .zip files, the container document should include the pertinent information for the K-map. If users navigate to it, they can open the attachment if they desire.

When filling out a Notes Data Repository form you have the option of turning the Notes spider loose on your database or refining exactly which documents it processes. Figure 6-3 on page 163 displays a Notes Data Repository form.
The View(s)/Folder(s) option allows you to limit the amount of data from the repository that gets spidered. You select the view or folders that contain the information you need. This choice is optional.

The Filter/Formula option allows you to enter a formula or filter to apply to either the entire repository or to the View(s)/Folder(s) specified. This choice is also optional. The format of the Filter/Formula must be in Notes formula syntax when the repository type is Notes. This field is editable after the repository has been queued for spidering (although, once it's queued, edits will not affect the current run). Check the “Process all documents during next run” box to apply the edits to the entire repository.

If your database contains a lot of small documents, such as “Thanks” documents that you typically find as replies in discussion type databases, then you don’t
want to include them. Here’s an example of a simple formula for a database that filters small and older documents:

```
SELECT (Form = "Main") & (@DocLength > 2500) & (createdate > [01/01/00])
```

**Tip:** If you are upgrading from a Discovery Server 1.0a or 1.1, implementation 2.0 will recognize the hidden ($DominoSpider) view previously used to refine documents selected from a database for spidering.

Some caveats for spidering Notes databases:

- Spiders collect access control lists (ACLs) for both repository and document-level access (if document access restrictions exist). The servers running the Notes spiders need Reader access to the repositories to spider and collect ACL information.
- If a document has a reader list that does not allow the Discovery Server that runs the spider access, the document will not be spidered.
- If the repository’s server is in a foreign Domino domain, you’ll have to enter that server’s name manually in the Server Name field on the Data Repository form, assuming cross-certification and connection records are already in place to allow connections to the foreign domain.
- Think about how your users access the documents. Some organizations use the Notes client exclusively, while others access Notes databases through a browser. You select a preferred viewer for each data repository you define.

**Lotus Domino.Doc databases**

The very same reasons that make Notes databases good candidates for Discovery Server spidering apply to Domino.Doc databases since they share the Notes lineage. Domino.Doc versions 3.0 and 3.1 are supported. When you define a data repository form for Domino.Doc, the following are true:

1. The Server option allows you to choose a server in the Server list box or type a server name in the text box and click the check box. Doing this populates a new Library list box with only the Domino.Doc library databases on the chosen server (based on the list in the ddadmin.nsf database on that server).

2. The Library option allows you to choose a library in the Library list box or type a library name in the text box and click the check box. Doing this populates the File Cabinet list box with only the cabinet files associated with the selected library.

3. The File Cabinet option displays all File Cabinets associated with the specified Library. Selecting one displays the file name in the Filename text box.
4. The Versions option includes the “Latest versions of documents only/Include all versions of documents” button. “Include latest versions of documents only” is the default.

**Lotus QuickPlaces**

QuickPlaces have quality metadata that can be leveraged with Discovery Server. Compared to a Notes database, they carry more design content. If your QuickPlace(s) contain lots of text, they are great sources of content for your K-map. Simply complete one data repository form for each QuickPlace and all its rooms will be spidered:

- The Server option allows you to choose a server in the Server list box or type a server name in the text box and click the checkmark button. Doing this populates the QuickPlace list box with only the main.nsf files from the QuickPlace directories on the selected server.
- The QuickPlace option displays all main.nsf files from the QuickPlace directories on the specified server, except for those found in the default QuickPlace, Help, and Tutorial directories. Selecting one displays the file name in the Filename text box.

**Windows-compatible file system files**

Windows-compatible file system files are good candidates for spidering if they were originally created with key information, such as author and title.

For example, do a quick audit of the properties of your MS Word files. If author information is missing, then the files won't be any good when you generate affinities later. If the title of the document is missing, then a lot of manual editing of your documents will be required in their native application so they can be properly categorized.

Fill out a data repository form for each public folder and all its subdirectories will be spidered:

- The Spider subdirectories option allows you to limit the subdirectories you spider. It's often a good idea to limit them at first so that you get an idea about the value of the data in the directories and subdirectories. Administrators are often surprised by the number of files in any one directory.
- Although the Discovery Server can access files on other file systems, like Novell or Samba, it can't get ACL information from these systems. In these cases, Discovery Server provides an override capability to allow you to spider content without ACL information. If the spider can't spider an unknown file system, you can select an ACL override on the Data Repository form.

Be sure to read about Taxonomy Import in the last section of this chapter if your organization has a well-organized file system structure—you have the option of importing it to help create your initial K-map.
**Web sites**

By viewing the source of Web pages you can determine if a site has enough metadata and text to be worthwhile. Web sites often contain lots of advertising. Frequency of content change on a site is another consideration. A news site, for example, changes its content throughout the day. Does it make sense to categorize it in one particular category?

If a site contains lots of links to other sites, it may not be useful, even though the Web Data Repository form gives you the option to “Spider Linked Pages”.

- The “Spider linked pages” option allows you to determine how much of a Web site you want to spider. It allows you to spider all links, none, or specify the number of levels to spider.

- The “Follow links to other servers” option is another way to create a subset of a Web site's data. Use this feature with discretion—you probably don't want to follow all the links on an Internet Web site.

**XML**

The new XML spider available with Discovery Server 2.0 gives you the flexibility to spider any source of XML data. For example, if your organization has a document management tool that allows for XML exporting, you can include that content in your K-map. Refer to the Architecture and Integration chapters for complete details.

**MS Exchange Public Folders and e-mail**

Spidering content in “Exchange public folders and e-mail” is now available with Discovery Server 2.0. For complete details see the Discovery Server product documentation.

To wrap up repository selection, remember that missing metadata is going to hamper your efforts. The better the metadata on your documents, the better Discovery Server will work for you and the less manual editing you will have to do of your K-map if you want a full-featured Discovery Server deployment.

### 6.3.2 K-map Builder settings

For your initial taxonomy creation we recommend leaving the K-map Builder settings at the defaults. Figure 6-4 on page 167 displays the choices that appear on the K-map Settings page before your initial K-map build.
The choices are:

- **Maximum number of documents to use when creating K-map**
  This setting only appears before you create your K-map. You can leave it blank or select a value of at least 3000. We recommend you set it at 3000 to 5000 and never more than 25,000 or you’ll run out of memory.

- **Maximum number of sub-category levels in K-map (depth)**
  This setting also appears only before you create your K-map. We recommend you leave it at the default since it controls how “deep” your initial K-map will be. The number of nested subcategories from the top or “home” of the hierarchy is created.

- **Limit how many categories a document can be automatically placed in to**
  As stated, this setting limits how many categories a document can be put in by the ATG. We recommend leaving this setting at the default value of 1 at least during the initial building of your K-map. Later in this chapter we discuss some options for the most efficient way to include documents in multiple categories.
Automatically create new subcategories if document maximum per category is reached?

This setting appears after your initial K-map generation and is used when you're ready to categorize your K-map. By selecting Yes, the Categorize feature of the ATG will add subcategories once you reach the maximum number of documents per category. Selecting No will instruct the ATG to add documents to the existing subcategories regardless of number. We recommend setting this to No for your initial K-map generation. When you are ready to turn categorization on, then change the setting to Yes to avoid the occurrence of any giant categories.

For both automatic and manual generation of subcategories, set the document maximum per category at

As stated, this setting limits how many documents can appear in a subcategory via either the ATG or manually using the K-map Editor. This setting also appears after your initial K-map generation.

Tip: When you change “Automatically create new subcategories if document maximum per category is reached?” to Yes, we recommend setting the document maximum in the next bullet to 500.

From the K-map Settings page you can also control who can edit your K-map. K-map Editors Group allows you to quickly add or remove the people you want editing your K-map. Again, we recommend that the people that you allow to work with your taxonomy are the content experts in your organization.

6.3.3 Field mapping

Lotus Discovery Server relies on data field maps to normalize field names in documents to the meta tags stored in the XML versions of the documents. It is not likely that all your data repositories will share the same field names; for example, some application designers might use an Owner or Created By field while others use Originator, all corresponding to Author (the meta tag in the field map). The $Global field map allows you to specify multiple field names that would accommodate the variety of references—in this case, tying them to Author. The Data Repository form displayed in Figure 6-3 on page 163 shows the $Global field map option.

In addition to the default $Global map, you might need to create one or more customized field maps for certain repositories. Unless you choose not to use the default map for a particular data repository, any new field map you create is combined with the $Global map for the repository. If you fully specify a custom map, you can completely override everything in $Global, but just for the repository it is tied to.
It's important that any new field map be specified before you initiate spidering. However, it is possible to associate a new field map with a certain data repository once the repository has been spidered. Just check the “Process all documents during next run” checkbox and it will be respidered completely. Figure 6-5 displays a blank custom field map form.

![Data Field Map - Microsoft Internet Explorer](image)

**Figure 6-5**  “Create new field map” option on Data Repository form creates a custom field map

**Stopword list**

Every Discovery Server installation leverages a set of language-specific, predefined stopwords.txt files – the stopword list. If you know that there are certain words in your organization that appear very often in your organization's documents and, for your users, are too common to be meaningful, you may want to add them to the stopwords list in order to increase the usefulness of the K-map. When the ATG task builds your K-map, it does not include any stopwords in your category titles and ignores them for clustering documents. The words in the stopwords.txt file are case-sensitive. We recommend adding your organization name to the stopword file, for example.
The stopword list is unique to every language and can be edited with a standard text processing utility like MS Notepad.

**Important:** In past versions of Discovery Server, administrators were warned not to enter the same word in the stopword list twice. This might happen with words that have different meanings but are spelled the same way and you don't remember having the word already added to the list. However, with Discovery Server 2.0 this is no longer a concern, as the ATG process simply ignores duplicate entries.

**Keywords**
Contrasting the stopword list are keywords. If you use a Field Map to map any field in any repository to the Keyword field, then the data in that field is given three times the weight of other words. For example, a keyword field on a particular Notes form may have been offered to help document creators choose a category for the information they were entering in one of your organization’s Notes applications. If this is the case, you can enter the field name in the Keywords field on the field map and the ATG will consider it 3 times as important as the words in the Title or Body field.

### 6.3.4 Generating your K-map and working with the K-map Editor

Generating your K-map is the moment you have been working toward. Keep in mind that your initial K-map can be recreated easily at this point because you have not edited it yet and turned on categorization.

For example, you may have to recreate your initial K-map if, in spite of your best efforts to choose quality content for spidering, a bunch of unusable characters appear. An organization may use codes or numbers to organize content in one of the repositories spidered. The ATG will dutifully categorize them but the codes may not make sense to everyone who will be using the K-map. To resolve a situation such as this, simply add the codes to the Stopword List and recreate your K-map.

If your organization is only implementing the search functionality of Discovery Server, then your work is done at this point. All you have to do is choose **Begin K-map creation** from the Startup page in the Discovery Control Center. Monitor the creation of your K-map, and when it is complete simply leave all categories generated by the ATG hidden. End users can now take advantage of your work by using a simple search UI such as the Atomica client (detailed in Section 8.5, “Complementary products” on page 244) or a custom UI that your organization has created with the Discovery Server API (detailed in Section 8.1, “User interface integration” on page 208).
Chapter 6. Developing and maintaining an enterprise taxonomy

**K-map Editor features**
Lotus Discovery Server Release 2 gives you better tools to edit and work with your taxonomy in the form of an improved K-map Editor and the eClassifier tool from IBM Research that ships with DS 2.0, which we discuss later. Figure 6-6 displays the K-map Editor followed by a list of the new and improved features in the K-map Editor.

**Important:** If you choose only to use the search functionality of Discovery Server, we highly recommend that you periodically check your K-map and follow the editing techniques discussed in the following section in preparation for the time in the future when your organization decides to leverage more Discovery Server features such as affinities, expertise profiling, or allowing users to navigate your taxonomy with the K-map UI.
The features are:

- **Fit Values** are now calculated on a scale of 0 to 100. Zero indicates that a fit value has not yet been calculated for a document. Fit value of 1 to 100 represents the strength of a document's correspondence to a category: the higher the number, the higher the correspondence. For more details on how Fit Values and Doc Values are calculated, see the Metrics chapter.

- Categories are now hidden automatically when created by the ATG. They appear greyed out in the K-map Editor and are invisible in the K-map UI. You decide when to “publish” a category. It will then appear in bolder type with a yellow icon in the Editor as shown in Figure 6-6 on page 171. The category “Appliances and Electrical” has been published.

**Note:** a parent category must be published in order for its siblings (sub-categories) to appear.

**Important:** Hiding a category does not prevent users from seeing the documents in that category if they are returned via a K-map search.

- Documents can be locked so the ATG can’t move them to another subcategory below the current category. All documents appear with an “unlocked” setting after the K-map is initially created. Locking becomes important in the latter stages of your K-map editing when you have refined it to the point that you want to anchor documents in categories.

- Document counts for each category are now visible.

- “Subdividing” of a category is now available. “Request Subdivide” will empty the parent category and create new subcategories underneath it. This feature does not occur immediately. The category icon changes to indicate a subdivide request has been sent to the K-map Builder service. The subdivide happens as soon as you (and any other editors in the K-map Editor) are idle or exit.

- Subdividing is the best way to work with your K-map. Do not spend hours manually moving documents around, let the ATG subdivide your large categories for you. Afterward you can work quicker with the resulting smaller subcategories. The ATG will flush out documents that are relevant and not relevant to the original category. You can then follow the tips in the Initial Taxonomy Edit section below to refine the results.
Tip: When doing a subdivide remember that the “Request Subdivide” feature subdivides the category at the sublevel below and clusters the documents without regard to the existing categories in your K-map (as the ATG does). Therefore, you will get the benefit of having like documents categorized together but you may still have to move and merge categories and/or documents manually.

- Retraining of a category is now possible by selecting Request Retrain. Retrain recalculates the Fit Values within a category. This feature also does not occur immediately. The category icon will change to indicate a retrain as a request is sent to the K-map Builder service. The subdivide will happen as soon as you (and any other editors in the K-map Editor) are idle or you exit.

- The K-map Editor now lets editors request two reports specific to their needs: Total Document Count by Category and Documents Added by Category. The first report is a great way to easily print out the structure of your K-map and the number of documents in each category. The second report allows you to see all documents that were added during a specified time period.

- The K-map Builder log has been enhanced and includes logging of more editor activities. This feature allows you to see the history of manual changes to your K-map for reference or audit purposes.

- All columns in the K-map Editor are immediately sortable to help you quickly reorganize the documents you are working with.

K-map Editor changes are “live” and happen online. The K-map Editor will give you occasional messages that the K-map service is running and you cannot do any edits. These messages have a duration of a few seconds, so choose OK for the first message and then simply wait for the second message to disappear. Don’t choose Exit. Figure 6-7 and Figure 6-8 on page 174 display the messages you’ll see in the order that they appear.

![Knowledge Map Editor Error](image)

*Figure 6-7  The first K-map busy message - click OK*
In an ideal world only one person at a time would be working in the K-map Editor due to its aforementioned “live” state. We realize, however, that your taxonomy team will most likely be comprised of content experts from different areas of your organization. In order to keep law and order in your K-map we recommend that you create a process for multiple K-map editors to work. For example, you could create a schedule of different hours for people to work in the Editor. You can also divide up the categories that people edit so they work only in their area of expertise. And remember, the K-map Editor logs all activity in the Editor so you can set up a process to audit it every time editing occurs.

The K-map Editor remains a Windows 32 bit application and must be run from your desktop while you are connected to your network. Direct the people you have chosen as editors in your K-map Editors group to download the new K-map editor from your Discovery Server at:

http://primaryserver.domain.com/kmapeditorinstall.html

**6.3.5 Initial taxonomy edit**

Your overall goal after the Automatic Taxonomy Generator (ATG) feature of Discovery Server has created your initial K-map is to shrink it down to make it as manageable as possible. Follow the recommendations in this section to maximize the time you spend editing by publishing the most obvious and relevant categories that the ATG created for you. Don’t worry about all the other categories—remember, they’re hidden.

Without doing any renaming of your main categories, expand your K-map so all subcategories are visible. Scan through the entire K-map and start to get a feeling for how things were categorized. The categories generated by the ATG...
have titles comprised of three words. The three words may look odd or unrelated but typically you'll find at least one word that represents what the category is all about. Figure 6-9 shows a freshly generated K-map with category names comprised of three words.

<table>
<thead>
<tr>
<th>Name</th>
<th>Fit</th>
<th>Status</th>
<th>Cat...</th>
<th>Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>appliance, n, p, off, expensive (112)</td>
<td>6/2</td>
<td>...</td>
<td>Def...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>bathwater, heat, dog (114)</td>
<td>6/2</td>
<td>...</td>
<td>Def...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cabinet, kitchen, contractor (137)</td>
<td>6/2</td>
<td>...</td>
<td>Def...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>house, paint, lat (106)</td>
<td>6/2</td>
<td>...</td>
<td>Def...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>base, panel, replacement (99)</td>
<td>6/2</td>
<td>...</td>
<td>Def...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>soil, garden, nursery (192)</td>
<td>6/2</td>
<td>...</td>
<td>Def...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sweep, roof, chimney (92)</td>
<td>6/2</td>
<td>...</td>
<td>Def...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>tile, plywood, board (142)</td>
<td>6/2</td>
<td>...</td>
<td>Def...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Uncategorized Documents (36)</td>
<td>6/2</td>
<td>...</td>
<td>Def...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 6-9  Initial generation of a K-map

Do not be overwhelmed trying to understand every category and subcategory that was created. Taxonomy editing can actually be a fun exercise, especially if you're the type of person who likes to create order in the world.

**General best practices for K-map editing**

Prior to starting any manual editing of your K-map, it is critical that you and any other taxonomy editors in your organization agree on naming conventions. As mentioned previously, it is important to have a taxonomy team that understands where your organization fits in the world. For example, regarding IBM, would you want to label your categories:

- IBM
- International Business Machines
- International Business Machines - IBM
- IBM - International Business Machines

We recommend one of the latter so that, no matter how your users utilize your K-map, they will be able to find the category by its acronym or full name.

The K-map Editor sorts documents based on their Fit Value. The K-map UI sorts documents based on their Document Value. It is important to be cognizant of these behaviors when editing your K-map. We recommend loading the K-map UI periodically to see how documents appear from an end user's perspective. You
can do this quickly in the Editor by selecting **View -> Knowledge Map Category** in the browser.

Smaller subcategories with few documents should be merged into the larger parent categories and subdivided later if desired. Using the K-map Editor, simply select all the documents in the subcategory and drag and drop them into the parent category, then use the Remove Category feature to delete the now empty category.

**Important:** A “small” category is any category that contains 10 or less documents. If you leave your categories small, chances are the ATG will not recognize their strength in the future regardless of the Fit Values of the documents.

Another option for eliminating small subcategories that works particularly well with large training sets is to create a temporary subcategory (we like to name it “KMap Maintenance”) under the main category and move documents there for later subdivision. This merging method is also helpful in the latter stages of your initial edit if you find documents that in your judgement don’t belong in the category they were originally placed, and are not related to any of the subcategories either. By consolidating them in a temporary placeholder you can subdivide the category after you’ve finished editing your K-map and are comfortable with the categories it contains.

**Important:** The more subcategories a parent category has, the lower the Fit Values will be in all the sibling categories. Never create more than 10 subcategories per parent category.

Moving entire categories around your overall K-map hierarchy is fine. This allows you to create your own high-level categories and design a hierarchy that makes the most sense for your organization. In other words, let the ATG create your categories, you create your hierarchy.

Move individual documents to other categories with discretion. We recommend retraining the category after moving to make sure the Fit Value remains high with other documents in the category.

Category Links are the most efficient way to include documents in multiple categories. It is certainly possible to manually copy documents to other categories or set the K-map Builder to increase the number of categories a document can appear in as outlined in 6.3.2, “K-map Builder settings” on page 166. However, including a document in multiple categories diminishes its Fit Value, and possibly others, in its categories.
When a category of documents appears to be relevant to another category in your K-map hierarchy, it is easy to create a Category Link using the K-map Editor. While dragging a category to its new category, hold the Ctrl and Shift keys until you drop it. The category will appear in its new, alternate location, but will simply be a link to the real category. Figure 6-10 shows a Category Link for “Hot water heaters” created under Heating. Notice the special icon and a Type of Category Link.

A Category Link will display in the K-map UI after a default refresh time of one hour. Figure 6-11 on page 178 shows the “Hot water heaters” Category Link in the K-map UI.
Steps to edit your K-map

1. Work from the “bottom up” by starting at the lowest subcategories.

2. Fit values over 50 are good. Generally, you will find very high fit values when editing your K-map. If a category is strong with documents that have very high values descending to documents that are below 50, that may be okay. Check out the documents by observing their titles or properties if necessary. However, if a category is created with a bunch of documents with fit values descending from 50, then it may be a weak one.

3. The highest fit values for documents in a category indicate what the category is truly about. Don’t be fooled by a bunch of documents that may appear at the bottom of the category with low values and a common word in their titles. By examining the documents in more detail, you will understand why the ATG placed the strongest documents together, and your human logic may determine that the weaker documents are either candidates to move or they are truly related.

4. Look at the titles of the documents to see whether there is a common theme in their titles. Often this will be enough to see some relationship.
5. If the titles of the documents don’t give you enough information to make a determination, look at the properties of each document by right-clicking it and selecting Properties. The Abstract box shown in Figure 6-12 provides a synopsis of the document:

![Figure 6-12](image.png)

6. Lastly, you can always double-click on the document and open it in its native application to see exactly what it is.

### 6.3.6 Categorization

When you and the rest of your K-map editors are comfortable that your K-map has categories that represent your organization’s content, then it’s time to turn on categorization. Categorization means that all new and modified documents will now periodically flow into your K-map. It is enabled via a checkbox on the K-map Settings form.

Enabling categorization causes your K-map to grow a bit depending on how long your initial editing period was and how many “small” documents were not included in the initial build. During the initial editing period of your K-map, the ATG is queuing up more documents to place in your K-map once you turn categorization on. You’ll also notice a certain number of documents appearing that were too small in terms of the number of tokens they carry, that is, they didn’t have enough tokens to match any categories in the initial build so they were kept...
aside by the ATG (in this context, small is not referring to the small size of a document in bytes as discussed in 6.3.1, “Choosing repositories” on page 161).

**Tip:** Categorization works best on “well balanced” K-maps that have somewhat proportional categories and reasonably proportional category sizes. K-maps that have categories with 3 to 9 subcategories are well balanced.

Avoid creating sibling categories with highly unequal numbers of documents. If you create two sibling categories with one much larger than the other, the larger category will attract more documents, no matter how high the fit values in the smaller category are.

Again, avoid creating categories with fewer than 10 documents. Combine the documents in these smaller categories with similar categories whenever you can.

If new documents do not end up in the categories you expect after turning on categorization, do the following:

1. Evaluate the fit values of the documents in the category. If the fit values are high, check the number of documents in the category and in its siblings. Balance the number of documents in the categories whenever you can.
2. Check the number of sibling categories. Create a higher-level parent category to avoid having too many siblings at the same level.
3. Move documents with fit values lower than 30 into more similar categories, or create a large temporary category that you can subdivide later.
4. Check the documents in the Un-categorized category. You may be able to create a new category if these documents are similar. If you do this, be sure to retrain this new category and evaluate the new fit values.
5. Subdivide any disproportionately large categories at any level. Make sure the Maximum Number of Documents setting does not exceed 500 documents. Set the K-map setting to automatically subdivide categories that have more than 500 documents to avoid large categories.

Categorization is an automated process provided to you by Discovery Server, but remember you have the tools and metrics to control what happens to your K-map:

- Don’t forget that you can lock documents so ATG will never move them.
- You can edit and retrain your categories periodically.
- There are new categorization reports available to both K-map editors via the K-map Editor and to administrators via the Discovery Control Center.
We recommend generating affinities prior to categorization just to make sure that your training set of documents created meaningful affinities. Conversely, we recommend spidering e-mail after turning on categorization to ensure matching of strong, populated categories in the K-map to spidered e-mail documents.

6.4 New tools for your taxonomy

After you have invested the time and effort to create a K-map for your organization, you are not limited in ways to maintain it or leverage it for your users. As you’ve learned in this chapter, the improved K-map Editor is certainly a great tool to use for keeping your K-map in tiptop shape. As you gain expertise, and new business needs come up, you can try the following additional tools to aid you in refining and maintaining your taxonomy.

6.4.1 eClassifier

eClassifier is a tool from IBM Research that offers an advanced editor with powerful features. It is currently not a fully supported product by IBM Software, and is therefore recommended only for advanced K-map editors. However, it has some excellent features that are not available in any other editing tool:

- Local/disconnected K-map editing
- Visual representations of documents and clusters
- Interesting metrics, such as cohesion and distinctness

This tool is described in more detail in Section 8.5, “Complementary products” on page 244.

6.4.2 Taxonomy Import

Taxonomy Import is a sample API that is included in the Discovery Server 2.0 API Toolkit. If, for example, your organization has an existing taxonomy in an operating system file system, you can import it into your K-map and get a jump-start by using it as the basis of your taxonomy. This can save significant time for sites that currently store a great deal of content in a well-organized and maintained file/folder structure. Your K-map can then serve as the frontend to your data repositories, helping users quickly find the documents they want, while looking at a familiar hierarchy. Equally important, the K-map can also point them to other documents in other repositories, creating categories of documents that extend far beyond the original folders upon which they were based.
Taxonomy Import also gives you the flexibility of creating a taxonomy in advance of your Discovery Server install. This option is of value for organizations who have an idea of what they’d like their taxonomy to look like.

Some considerations when using Taxonomy Import:

- Include at least 10 (preferably 15 to 20) representative documents in each folder.
- Be flexible in allowing Discovery Server to create additional categories. One of the major strengths of Discovery Server is its ability to quickly read and analyze many thousands of documents, and find connections and affinities you may not have thought about—in essence, tell you things about your corporate data you may not have known.
- Creating a new file system taxonomy to import is a way to jump-start your K-map. Don’t devote the rest of your life trying to gather all your documents from computers scattered throughout your site, figure out what’s in them, create categories accordingly, and then stuff each one into the correct slot. That’s Discovery Server’s job. The file system structure should only be used to create K-map categories. You can later populate these categories with documents using Discovery Server itself.
- You need the Java 1.1.8 JDK to run the Taxonomy Import program.
- You need to add yourself to the Administrators, KmapEditors, KDSAdministrators, and DiscoveryServers groups to run the Taxonomy Import program.
- You need to be sure that the File System spider is enabled on your Discovery Server.
- The Taxonomy Import program is run from the Discovery Server command prompt using a number of switches that are documented in the sample file.
- The same rules detailed in the preceding sections apply to an imported taxonomy. Make sure you have balanced categories, a good number of sibling categories, etc.

The Taxonomy Import program taps into the existing Discovery Server processes. First, it creates a temporary category in your K-map to hold all imported documents. It then creates a Data Repository record that you can view in the Discovery Server Control Center. You can monitor its progress by looking at this record. You can also watch the output that appears in the command prompt window.

The Taxonomy Import API can also be used to import a “branch” into an existing K-map. For example, as your K-map matures, more parties in your organization may want to have their information included. If it doesn’t fit in nicely into your original top level hierarchy, you can import a new branch.
6.5 Conclusion

By following all the tips and techniques in this chapter, you will be able to create an effective K-map for your organization. Remember to use the power of Discovery Server to your advantage when working with your K-map. Don’t forget the tips we’ve given you:

► Don’t fall into the habit of only manually editing your K-map. Let the ATG subdivide for you, then do your manual edits. As time passes you’ll begin to realize how efficiently you can work with Discovery Server.

► You’re in complete control. Don’t like a category? Leave it hidden until you decide to publish it later. If you publish it and get feedback from your users that it’s incomplete or inaccurate, simply hide it again until you get a chance to spider more repositories or edit it to clean it up.

► Don’t want your documents moved once they’ve been categorized? Lock them.

As you gain experience working with your taxonomy, you’ll see how efficiently you can spider more repositories to make your K-map an increasingly valuable organizational asset.
Chapter 7. Metrics and affinities

This chapter describes how Discovery Server collects Metrics about the information you ask it to spider. It explains what is collected, how it is processed, and how the metrics values are displayed via the Discovery Server user interface.
7.1 Metrics

7.1.1 Definition of metrics

“Metrics” is a Discovery Server process that evaluates documents, people, and categories of information, and then determines the relationships between them. Metrics consist of two basic processes: Metrics Collection and Metrics Calculations. The Metrics Collection process gathers usage statistics for data within the sources you spider, and the Metrics Calculations process calculates relative data values based on these statistics. These statistics, and the calculated values, are known as metric values.

For example, Discovery Server understands that there is a relationship between a document it spiders and the author of that document as recorded in the metadata for that document. If it can match the author against somebody whose profile is listed in Discovery Server profiles, it builds an association between that profiled user and that document, and can record an affinity between the author and the associated category.

Usage statistics are the raw data that is used to calculate a number of values that are recorded for people, documents and categories. As a result, the better the recording of usage statistics, the better the calculated values.

Some of the statistics are recorded by Discovery Server, as a result of user interaction within Discovery Server. But many of the usage statistics come from the documents themselves being spidered. For example, one of the usage statistics is the editors of a document. Because documents are edited in their native application outside Discovery Server, this data is only recorded by the Metrics Collection task as a result of the information obtained about that document's edits via the content spidering process. This approach is how Discovery Server records the editors.

In Notes documents, the $UpdatedBy field exists in all documents and can be used by the spider to record the edit history by person for that document. For word processing documents, this is also recorded, but is only meaningful if you have set up your word processing software to identify users when they use their word processor to edit a document.

This emphasizes a theme you will see throughout this book. Good metadata, the attributes of a document, is important in gaining full advantage of Discovery Server. If you don't have good metadata today, we strongly recommend that you put in place policies and procedures, and perhaps encouragement for your users, to build a good set of metadata as new documents are created and edited. You may decide that the older documents will not be updated, but you should definitely do this for all new and amended documents.
7.1.2 What metrics are collected

Metrics raw data is collected from source documents in a number of ways.

For all data sources except file system files, field maps are used. Field maps are used to define to Discovery Server the relationship between a field that is known and used by Discovery Server (for example, Author) and the corresponding field in a source document that has the same meaning, but may not have the same name (for example, WrittenBy).

For File data, the Keyview filters used to spider file system documents automatically extract known metadata and map it to relevant Discovery Server fields.

Key field mappings used by metrics
Table 7-1 lists the fields that are used by Discovery Server for the purpose of metrics. There are more fields in the field maps, but these have little significance for metrics (for example, Title).

Table 7-1   Fields used by Discovery Server for metrics

<table>
<thead>
<tr>
<th>Field</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Author</td>
<td>Used for affinity mapping</td>
</tr>
<tr>
<td>Date Created</td>
<td></td>
</tr>
<tr>
<td>Revision Date List</td>
<td></td>
</tr>
<tr>
<td>Revised by List</td>
<td>Used for affinity mapping</td>
</tr>
<tr>
<td>Date Last Read</td>
<td></td>
</tr>
<tr>
<td>Directed to</td>
<td></td>
</tr>
</tbody>
</table>

By ensuring that the field map correctly identifies the relationship between source data fields and Discovery Server fields, you allow Discovery Server to build meaningful metrics.

Action and Document constant metrics
Table 7-2 on page 188 and Table 7-3 on page 189 give a list of the metrics that are collected by Discovery Server. There are two types of metrics:

- Action constants - actions that are taken by users on a document, used for affinity calculations
- Document constants - recorded activity attributes for individual documents, used for document valuations
The meaning of these metrics will vary depending on the type of data repository spiedered.

Table 7-2  Action constants

<table>
<thead>
<tr>
<th>Metrics collected</th>
<th>Description</th>
</tr>
</thead>
</table>
| Authoring documents in a category    | The author field in the source document is used to record authorship in Discovery Server. Metrics in Discovery Server counts the number of documents authored in a category for each person Discovery Server has listed in the profiles.  
This metrics has most meaning for Notes documents which record authors automatically. For File and Web sources, it will depend on whether you have author information in the document metadata. 
Keyview filters used by Discovery Server spiders for files automatically read any metadata, and for HTML, metadata tags must be present. |
| Editing documents in a category      | Similar to the authored metric, this has most meaning for Notes documents, with File and Web requiring appropriate metadata. Keyview filters used by Discovery Server spiders for files automatically read any metadata, and for HTML, metadata tags must be present. |
| Linking to documents in a category   | For Web and Notes documents, links are evaluated. This metric is not evaluated for File documents.                                           |
| Responding to documents in a category| Only applicable to Notes documents.                                                                                                          |
| Reading documents in a category while using K-map | Applicable to all documents.                                                                                                                |
7.2 How metrics are used

This section explains how the usage statistics outlined in 7.1.2, “What metrics are collected” are used to make value calculations. Value calculations are made to produce two kinds of metric values, document metrics and people metrics, also known as affinities or expertise mapping.

7.2.1 Document metrics

The Discovery Server uses metric values to rank the data it processes. For example, when users browse categories with K-map, documents that have the highest values appear first by default in each category so that they’re easier to access. Once enabled, the Metrics system keeps the Discovery Server current by continuously updating metric values for existing data and calculating values for newly spidered data.

In the Metrics Settings form, you make the decision on the ranking of these constants for your organization or data sets to be spidered by Discovery Server. Discovery Server uses the relative importance of constants, as specified by you, to calculate metrics values.

The default order, based on the assumption of a variety of data sources, is as follows:

1. Links to a document

---

<table>
<thead>
<tr>
<th>Metrics collected</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Links to a document</td>
<td>For Notes and Web documents, links are evaluated. Links are not evaluated for File documents. All document data in Discovery Server is evaluated to return this metric.</td>
</tr>
<tr>
<td>Links for (in) a document</td>
<td>For Notes and Web documents, links are evaluated. Links are not evaluated for File documents.</td>
</tr>
<tr>
<td>Responses to a document</td>
<td>Only valid for Notes documents.</td>
</tr>
<tr>
<td>Times a document has been opened using the K-map</td>
<td>Valid for all documents.</td>
</tr>
<tr>
<td>Recency of the last update to document</td>
<td>Valid for Notes and File documents, Web documents dependent of recording of metadata.</td>
</tr>
</tbody>
</table>
2. Links from a document
3. Responses to a document
4. Number of times a document has been opened in the K-map
5. Number of times a document has been edited

For each of these metrics collected, Table 7-3 on page 189 lists considerations for each type of data source that is being spidered. Consider these when deciding the order of metrics values that are appropriate for your organization.

If you decide to change the order of these metrics weightings, from most important to least important, you must understand that the same order is applied to every repository that is spidered. You cannot assign an order by repository. As a result, there are only special circumstances where you would change the default order.

As an example, if you decide to set up a Discovery server to spider external data, for example public Web sites for industry-specific information, the authors of the documents are likely to have little significance. This is because they would not be listed in your directory and so no affinities would be assigned. However, the number of times a document is opened from the K-map metric might be considered a valuable metrics to track, for document value.

### 7.2.2 Metric calculation

In each list of constants, Metrics assigns the first constant a value of 1 and reduces the value of each successive constant by 30%. For example, when the Action constants are in their default order, Metrics assigns the first constant (Authoring) a value of 1, the second constant (Editing) a value of 0.7, the third constant (Linking to) a value of 0.49, and so on.

If you change the order of a list, each constant assumes the value associated with its new place in the list. For example, if you move the Editing constant ahead of Authoring in the Action Constants list, Editing assumes a value of 1 and Authoring assumes a value of 0.7.

Metrics applies the values of these constants to the raw metrics it collects to calculate affinity and document values. For example, the following table shows how a sample document's value is calculated (assuming the document constants are in the default order indicated in the preceding section).
Table 7-4  Sample calculation

<table>
<thead>
<tr>
<th>Raw metric</th>
<th>Constant value (from Settings form)</th>
<th>Calculated value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of links to document = 10</td>
<td>Links to = 1</td>
<td>10 x 1 = 10</td>
</tr>
<tr>
<td>Number of links from document = 5</td>
<td>Links from constant = 0.7</td>
<td>5 x 0.7 = 3.5</td>
</tr>
<tr>
<td>Number of responses to the document = 3</td>
<td>Responses constant = 0.49</td>
<td>3 x 0.49 = 1.47</td>
</tr>
<tr>
<td>Number of times the document has been opened with K-map = 23</td>
<td>Times opened constant = 0.34</td>
<td>22 x 0.34 = 0.68</td>
</tr>
<tr>
<td>Number of times the document has been edited = 4</td>
<td>Recency constant = 0.24</td>
<td>4 x 0.24 = 0.96</td>
</tr>
</tbody>
</table>

** See Calculation notes for an explanation of this value

Metrics then adds the individual calculated values together to produce a document's total value; in the sample document's case, the total value is 16.61. However, if you switched the Links to constant to the third position and the Responses constant to the first position in the Documents Constants list, the sample document would have a value of 11.14 (a reduction in value of about 33%). This example suggests the effect the order of the constants can have on Metrics calculations.

**Calculation notes**

Metrics reduces the number of times a document has been opened with K-map by the number of responses to the document so that implied actions are not counted twice. (Metrics assumes that the author of a response has opened the original document at least once.) In the preceding example, Metrics reduced 23 (the Times opened raw metric) by 3 (the Responses raw metric) to produce a new metric value of 20.

Metrics divides the number of times a document has been opened by 10 before it applies the Reading constant in the Action Constants list or the Times opened constant in the Document Constants list. Metrics does this to prevent frequently-opened documents and people who open a large volume of documents from having disproportionately high document values and affinity values, respectively. In the preceding example, Metrics divided 20 (the Times opened raw metric; see previous paragraph) by 10 to produce a new metric value of 2, which is the value to which Metrics applied the Times opened constant.
7.3 People metrics (affinities)

People metrics are referred to as affinities or expertise. These metrics calculate attributes of people who are listed in the K-map, such as people who have authored or interacted with spidered content. It then uses these attributes to assign a strength of associations, called affinities, with categories that Discovery Server catalogues.

The affinities are used to show people who have a calculated affinity for a category or search results, giving you a “who knows about this” in addition to the “what is there about this” result of document browsing or search.

7.3.1 What affinities are

Affinities are calculated using the five action constants listed in Table 7-2 on page 188. As with document metrics, in the Metrics Settings form, you make the decision on the ranking of these constants for your organization or data sets to be spidered by Discovery Server. Discovery Server uses the relative importance of constants as specified by you, to calculate metrics values.

The values of these five constants are used in the following priority to determine the affinity of a person to a category or search result:

1. Authoring documents in a category
2. Editing a document in the category
3. Linking documents to category
4. Responding to documents in a category
5. Reading documents in a category while using the K-map

For each of these metrics collected, Table 7-2 on page 188 lists considerations for each type of data source that is being spidered. Consider these when deciding the order of metrics values that are appropriate for your organization.

As for document constants, you are not likely to change the default order, unless you have a particularly specialized set of data on your Discovery server.

Affinity assignment

The affinities calculated by Discovery Server are assigned according to the Affinity Generation Threshold in the Affinity settings form of the Discovery Server Control Center.

This threshold is expressed as a percentage, for example, 65%. This means that a person will have an affinity assigned or proposed if they have a greater affinity
value for a category than 65% of all the people that have an affinity to the category.

To give a further example, 100% would mean that only one person would be assigned an affinity. Alternatively, a strength of 1% would calculate many individuals having a fairly weak affinity to the category term, which is not useful to organizations seeking experienced and knowledgeable category/topic experts. This threshold is a Discovery Server global value for all data spidered by Discovery Server.

Lotus has found that a threshold between 55% and 65% gives a useful assignment of affinities.

**Affinity publication**
This is done in one of two ways:

- Automatically assigned and publicized by Discovery Server
- Accepted by a person for whom the affinity is being proposed for publication to his/her profile

You can specify in the Affinity settings form of the Discovery Server Control Center (see Figure 7-1 on page 194) that the user will be asked to accept proposed affinities.
If you deselect the checkbox “Send e-mail requests to end users for approval of proposed affinities”, affinities will be automatically assigned and published.

This setting is sometimes used during a pilot or test of Discovery Server, but is generally not used during Discovery Server production deployments due to data privacy policies in place within organizations. Which method you choose is an organizational decision, and should be considered as part of the deployment planning.

**Designated and declared affinities**

There are two additional ways that affinities can be assigned to a person:

- Explicitly assigned to another person’s profile by a member of the AffinityDesignators group defined on the Discovery Server, designating that person’s expertise in a category.
Let's look at some usage examples.

You would use the first case where you have document authors who are not the subject matter experts. In this case, Discovery Server would identify the author as having an affinity, but not the subject matter expert.

The second case would be used by a person who is new to the organization and so has not contributed much yet, but is a subject matter expert in one of the category areas managed by Discovery Server.

It is important to note that if one self-declares a category to their profile, they will initially receive a lower-ranked affinity score than those that may be discovered automatically by Discovery Server. However, Discovery Server will continue to monitor their work for increased affinity score valuations to this category.

### 7.3.2 How affinities are shown by Discovery Server

Affinities can be seen in a number of areas of the Discovery Server user interface.

If you do a search with Discovery Server, the results include people who have an affinity to the topic or categories found. Figure 7-2 on page 196 shows an example.
Whenever you see a people name, you can click the name and choose the Show Profile to see all affinities for a person on the Affinities tab of the profile. Figure 7-3 on page 197 shows an example.
When you browse the K-map, you will see a list of people who know about sections of the K-map for each category you open. Figure 7-4 on page 198 shows an example of this.
Figure 7-4 Browse K-map

In each of the figures above, you also see the integration of IBM Lotus Sametime, to provide online awareness. Carol Easthope is online (square besides her name), Susanna is away (diamond) and Brian is on Do-not-disturb (circle with line). See Section 8.5.3, “IBM Lotus Sametime” on page 249 for more information on IBM Lotus Sametime integration.

7.3.3 The use of e-mail in generating affinities

The widespread use of e-mail provides a valuable source for understanding the business interests and knowledge of people in your organization. The philosophy behind Discovery Server’s ability to evaluate e-mail content is to attempt to raise affinity scores for individuals who do significant work within a Kmap category topic in e-mail.
Your users will naturally be concerned about the concept of e-mail being evaluated by Discovery Server to find out what people know. So it is important to understand and communicate how e-mail is used by Discovery Server, should you decide to make the e-mail spidering option available to your users.

Information spidered from e-mail is only used for affinities evaluation to existing K-map categories, and does not lead to the creation of new categories based on text within the e-mail. E-mail documents do not appear in the K-map, nor are they stored, except in a temporary encrypted work queue (see Chapter 2, “The Discovery Server architecture and key components” on page 13 for further details on the e-mail spider).

So, the fact that you have an affinity for shopping or fishing, determined from e-mail, will not lead to a proposed or assigned affinity to fishing or shopping, unless the category already exists in the taxonomy; nor will your documents discussing fishing and shopping show up in the K-map.

**How e-mail is used in affinity calculation**

The e-mail spider records the From, To and Carbon Copy (CC) fields from each e-mail so that it can give affinity score credit to the e-mail authors and some credit to recipients as well. It only records individual names that it can match against the profiles of Discovery Server. Group names and names not listed in the profiles of Discovery Server are ignored.

The content of each mail message is used to calculate affinity values for existing categories already in the K-map, for each individual.

Whether an affinity is proposed for an individual depends on the Affinity Generation threshold defined in the Affinity Settings form of the Discovery Server Control Center (see “Designated and declared affinities” on page 194”).

**Privacy considerations**

The privacy of e-mail is fully protected by Discovery Server. Nevertheless, organizations are encouraged to review their corporate data privacy policies, as well as local, international, and industry-specific privacy regulations that may apply to their use of electronic data. For example, some countries explicitly forbid any examination of e-mail.

If you are able to spider e-mail for affinities, we strongly recommend that you do so. E-mail is an invaluable source for knowledge and experience.
7.3.4 Affinity changes

In addition to the facility for an administrator to remove an affinity from a user's profile, affinity changes also happen automatically over time.

Discovery Server examines the affinities of people to categories over time and, if appropriate, will reduce the affinity of a person to a certain category, if that person has less to do with that category. This process happens in the following manner:

If a person has an affinity to a category and that affinity is not changing, the affinity of that person to the category will reduce (decay) by 1% of its value every day (for example, 100 to 99 to 98.01 to 97.02 etc.). This only happens if other affinities (for other people) to that category are updated. If no affinities are changing in the category, all of them stay the same. The end result is that people who are regarded as "active" in the category (for example, through document ownership or activity) remain with a high affinity, while inactive people slowly have their affinity reduced.

As soon as the person has a change to his/her affinity to that category, the decay is removed and they regain their original affinity rank. This is done because Discovery Server doesn't really consider them to have "lost knowledge"—they are just less active. As soon as they are active in this area again, they "deserve" their prior score.

The score bottoms out at 1 because Discovery Server never removes affinities from categories unless people decline them, or an administrator removes them from someone's profile.

7.4 Metrics reporting

People metrics (affinities) and document metrics are shown in various places in the Discovery Server user interface, as outlined in 7.3.2, "How affinities are shown by Discovery Server". In addition to document and affinity values displayed in the Discovery Server user interface, individuals with administrator access to Discovery Server can use the metrics report generator, available in the Discovery Server Control Center, to generate on-screen reports.

Standard reports

The standard metrics reports available in the Discovery Server Control Center are as follows.

► Top authors by document value
► Top categories by activity
► Top documents by access
Top documents by link source
Top documents by link target
Top document by responses
Top documents by value

Custom reports
In addition to standard reports available with Discovery Server, it is possible to create custom reports.

Figure 7-5, and Figure 7-7 on page 203 show the parameters available for custom reports for categories, documents and people.
Figure 7-6  Documents
Metrics reporting using the Discovery Server API

It is also possible to use the Discovery Server API to access metrics information in Discovery Server. You can use this to create your own custom reports, or to use the retrieved information in other systems. You may wish to provide some reports to your users via an intranet page or to create an automatically printed report on a periodic basis.

The Discovery Server API provides a sample, MetricsSample, that uses the LDSMetrics, LDSMetricsQuery, and LDSMetricsResults classes to extract metrics information from Discovery Server.

More information on the Discovery Server API can be found in Section 8.4, “Integration using Discovery Server programming interfaces” on page 231.
7.5 Metrics backup

Discovery Server provides for backup and deletion of metrics collected. It is important to note that Discovery Server only deletes (if specified) the raw metrics data collected, not any calculated metrics that are used for document value or affinities.

The intent for the metrics backup and metrics restore procedure is to remove the “raw” metrics from the database after they have been processed by Affinity Processing and Document Value Processing so that the tables will not grow without bounds with data that isn’t going to be used again.

The option to delete and/or back up raw metrics data is provided in the Metrics Collecton section of the Discovery Server form, for the server on which the Metrics Collection service is running. You access the server from the Discovery Server Control Center.

By default, raw metrics are deleted after 90 days and no backup is done. You can change the delete period and whether to back up before deletion.

Metrics backup is not intended as a data recovery strategy. You must use conventional backup tools, such as Tivoli Storage Manger, to perform scheduled backups for disaster recovery.

Metrics backup is primarily used in conjunction with the Metrics restore capability of Discovery Server. A restore capability is provided to allow you to put back deleted metrics without doing a full system restore (and wiping out other states in the same tables) for the purpose of doing reports or some other query using the API.

The backup and restore capabilities only process the subset of the metrics that are considered to be “raw” metrics, not the affinity or document value metrics that are calculated.

7.6 Conclusion

Metrics, and the resulting document values and people affinities, is one of the most valuable tools that Discovery Server offers your organization, because they give you a sense of value about information and people in the organization.

It is important to see them as a complement to the other ways in which people gather and value information and contacts. They must never be used in isolation or as an absolute measure of the value of documents or people in your organization.
It is also important to prepare your organization for the introduction of such measures so that your people understand what they mean in the context of the usage of Discovery Server. This topic is covered in Chapter 3, “Planning for a successful deployment” on page 59.

Lastly, give strong consideration to spidering e-mail for affinities.
Integrating Discovery Server into your enterprise

Discovery Server is a tool that will be used in conjunction with other systems in your organization. This chapter provides an overview of the ways in which Discovery Server can be integrated with other systems.

The following topics will be discussed:

- User interface integration
- Directory integration and Single Sign On (SSO) considerations
- Non-supported data sources via the XML spider
- Discovery Server Application Programming Interface (API)
- Complementary products

This chapter is not a step by step guide to achieving integration for any of these purposes, rather provides an overview of what is possible and how it works. It should be read in conjunction with existing product documentation, and knowledge of programming environments such as Java and XML is assumed.
8.1 User interface integration

The major capabilities of Discovery Server – search, profiles/affinities, and K-map categories – will very rarely be used on their own. In many cases, organizations will want to make the capabilities part of existing applications, intranet sites, or portal applications; rather than simply accessing them through the standard K-map interface provided. Discovery Server provides a rich set of interfaces that can be used to easily integrate into existing or planned applications.

While your ultimate aim in using Discovery Server may be to use the standard K-map browsing interface, we recommend you start by making some of the basic Discovery Server capabilities and user interface elements available as integrated parts of existing Web pages. Taking this approach allows you to go through stages in deploying the facilities of Discovery Server, without user training for the full K-map browsing capability.

You might think of the various user interface options as being a “lite” version of the built-in K-map user interface. Such alternative user interfaces are also important if you do not use Microsoft Internet Explorer, since Internet Explorer is the only supported browser for the full Discovery Server K-map interface.

In order to integrate Discovery Server capabilities into your existing Web applications, you will need to use a number of functions of the Discovery Server API. An overview of the Discovery Server API is presented in 8.4.1, “The Lotus Discovery Server API Toolkit” on page 231, including a discussion of the API samples that are included in this toolkit to get you started.

Some of the possible interface options are demonstrated in the following sections.

8.1.1 A simple Discovery Server search interface

Search is likely to be the first, and most used, interface you will want to provide for your users. A simple search interface can be used to provide the searching of documents, people, and/or categories - while not requiring navigation, browsing, and understanding of a corporate taxonomy. Search capabilities are available from Discovery Server after it has been simply spidered and indexed, even prior to the completion of the K-map category structure.

Figure 8-1 is an example of a simple Discovery Server search results screen.
We used the K-map search servlet sample from the Discovery Server API toolkit to create the example shown in Figure 8-1. After compiling the source code, the servlet was installed on the Discovery server, as per instructions available in Discovery Server API documentation. The servlet is then called by this URL:

http://itsolds1.lotus.com/servlet/KmapSearch

In this example we chose only to return categories and documents, but people could easily be returned in the results with some modifications to the sample servlet. Other modifications that might be considered would be to add Sametime people-awareness capabilities, as discussed in 8.5.3, “IBM Lotus Sametime” on page 249.
8.1.2 A simple K-map browsing interface

Out-of-the-box, Discovery Server provides a full K-map browsing and search interface, called by the URL yourserver.yourdomain.com/kmap.htm. This interface is very feature rich, but may not meet your performance, browser limitations, or other needs. Additionally, this interface may provide more options than are necessary for your application purposes.

Therefore, you may wish to start users with a simpler version of the K-map browser once they have graduated beyond the basic search interface.

Figure 8-2 shows an example of such an interface.
installed on the Discovery server, as per instructions available in Discovery Server API documentation. The servlet is then called by URL:

http://itsolds1.lotus.com/servlet/KmapBrowse

In this example we chose to return categories, people, and documents.

8.1.3 Utilize the full K-map interface

After considering the other interface options, the full out-of-the-box K-map interface may be desired. In this case, it is generally a good idea to replace some of the default graphics utilized in this interface. For example, you may want to replace the K-map logo with your corporate logo, or change the background and other colors to better match your corporate intranet.

Complete instructions for modifying the full K-map interface graphics and color themes can be found in the product documentation, within the Lotus Discovery Server Control Center Guide (ds_admin.chm).

For help accessing the Control Center Guide, see “Related publications” on page 297.

8.1.4 A summary of Discovery Server interface options

As mentioned previously, with Discovery Server 2.0 there is no need to utilize the full K-map browsing interface if you find that it does not meet your needs. There are many interface options that you can choose from. They are described briefly in this section, and the pros and cons of using each are listed.

Basic search
The basic search interface is based on the search servlet sample in the API toolkit.

Pros

- Basic search provides users with a quick and easy way to leverage Discovery Server and search indexed content.
- Since taxonomy is hidden to users, there is less effort required to get started with Discovery Server. The auto generate ATG taxonomy can often be used, and more time can be taken by taxonomy editors to refine the taxonomy structure prior to publication for users.
- It can be fully customized to your existing Web look and feel, including support for multiple Web browsers.
Cons

- Basic search does not take advantage of all of the strengths of Discovery Server.
- It requires development effort to harden and customize the API sample, as well as to maintain and update it for future releases.

**Basic K-map**
A basic K-map interface is based on the K-map browser servlet sample in the API toolkit.

Pros

- It provides a good stepping stone for advanced usage of Discovery Server, as not all the full K-map features need be exposed.
- It can be fully customized to your existing Web look and feel, including support for multiple Web browsers.
- It can provide a higher-performance option over the out-of-the-box full K-map interface.

Cons

- Basic K-map may not take advantage of all of the strengths of Discovery Server.
- It requires development effort to harden and customize the API sample, as well as to maintain and update it for future releases.
- The taxonomy is more exposed than in a simple search interface, so it requires a more clearly defined taxonomy.

**Full K-map**
The full K-map browser interface was described earlier in this chapter.

Pros

- It fully leverages all Discovery Server capabilities, providing a very rich interface.
- No development effort is required to utilize this interface: Lotus software will maintain the interface, updating it with each new release of Discovery Server.

Cons

- The full K-map interface only supports Internet Explorer.
- It is limited in its ability to be customized to your existing Web look and feel.
- It requires a clearly defined and well edited taxonomy, as this is the primary navigation tool in this interface.
- The feature rich UI will be a bit slower than a simpler interface.
Custom
You can create a new interface from scratch via the API toolkit.

Pros
- The interface can be fully customized to your existing Web look and feel, or to integrate with other applications, such as legacy systems, erp, and so forth.

Cons
- A custom interface requires development effort, possibly a significant amount, depending on the features desired.

Third-party
There are several complementary products that can provide an interface (see 8.5, "Complementary products" on page 244).

Pros
- Complementary products (for example, Atomica) can sometimes provide other valuable features in addition to accessing all the Discovery Server capabilities.

Cons
- This approach may require additional software purchases, as well as significant additional deployment and planning efforts for the additional technology.

8.2 Directory integration and single sign-on

Directory integration is an important and often difficult topic to consider when deploying any new application to your enterprise infrastructure. In most cases, directory integration issues come down to concerns about authentication (does the user have access to the system) and authorization (what can the users see and do once they are in?).

8.2.1 Authentication concerns

This area of concern includes verifying who the user is, and whether they are actually allowed access into the system. The traditional approaches here are either to use an authentication directory specific to your given application, or to trust an external LDAP or other authentication directory.

Discovery Server allows either of these options to be utilized since it is built on the underlying support of Lotus Domino. It can use the Domino Directory as its
When integrating the Discovery Server platform into your corporate intranet, you may have a corporate homepage that users initially authenticate with, from which they will link to Discovery Services. If you do not want them to be prompted again for authentication when accessing Discovery Server, you can implement a single sign-on solution that allows Discovery Server to trust the authentication trusted by your corporate homepage or other referring source.

Again, since Discovery Server is built on top of the underlying support of Lotus Domino, it can leverage the SSO support built into the Domino product.

**SSO with other IBM technologies**

Single sign-on with other Lotus/IBM technologies can be enabled via use of the Lightweight Third-Party Authentication (LTPA) technologies included in Domino and IBM WebSphere-based products. For more details on LTPA token setup, see the Discovery Server Control Center Admin Guide (ds_admin.chm) or the redbook *Domino and WebSphere Together*, SG24-5955. These sources discuss the various domain/hostname, realm, and token expiration issues that can often plague LTPA setups.

**SSO with non-IBM technologies**

If single sign-on with non-IBM technologies is required, then there are two options.

First, you can create a custom DSAPI filter to handle authentication against an external directory or other authoritative source. The Domino Administrator documentation available from the Lotus Developer Domain website (www.lotus.com/ldd) can provide more information on developing such DSAPI filters.

An example here would be to have the corporate homepage create a proprietary cookie “token,” which would certify their authentication credentials, and then write a custom DSAPI which decrypts and utilizes the information within this cookie to accept the authenticated users into Domino/Discovery Server.
A second option for providing SSO capabilities with non-IBM technologies is to utilize a centralized security management application, such as Tivoli Access Manager for e-Business (formerly known as Tivoli Policy Director).

Such “access/security manager” products integrate with many applications, application servers, and portals; they typically offer C and Java APIs that can be used by custom security applications. To protect HTTP-based applications such as Discovery Server, they function as a Web reverse proxy security server. A reverse proxy server is a Web server that listens on an HTTP port, typically 80, for requests intended for particular URLs, shielding those URLs from direct access. Thus, a reverse proxy “security” server can provide authentication and authorization before forwarding a request. In our example, both the corporate homepage and Discovery Servers would trust this access/security manager to handle security on their behalf.

For more information on the Tivoli Access Manager, which has been tested for use with Discovery Server deployments, see the Tivoli website at www.tivoli.com, or the redbook Enterprise Business Portals with IBM Tivoli Access Manager, SG24-6556.

### 8.2.2 Authorization concerns

Once a user has been successfully authenticated into the system, the next question is what they are “authorized” to see and do.

**Person profiles**

When considering the first part of this equation, what the user can see, Discovery Server handles a good amount of this for us. As discussed previously (2.5.11, “Repository security, and spider ACL collection” on page 35), Discovery Server collects and honors the access-control rights of all data repositories spidered. This means that when a user searches or browses the Discovery Server K-map, they will only see document metadata for documents to which they have access.

However, there is one catch to ensuring that this process works correctly: ensuring proper person “profiles” are gathered by Discovery Server, including information about the user names for various content systems. Person profiles are created and maintained by Discovery Server as a single collection point for information about users pulled from various authoritative (LDAP and Domino Directory) and supplementary (teamroom and discussion database) sources, and they are regularly populated with affinities to categories in the K-map. For affinity generation, and for document ACLs to be monitored, it is important that users are properly mapped between the multiple profile sources.

For example, if a user logs in as Bob, then there needs to have been a corresponding “Bob” person profile generated by the profile spider. This way, any
Notes document in the K-map that has access limited to a specific group, of which Bob is a member, Bob will be able to see. However, Bob may be known as “Robert” to a Microsoft Exchange repository that has also been spidered. Therefore, for Bob to see the Exchange documents limited to “Robert”, “Robert” must be listed as an alternate name for “Bob”.

More details on choosing profile sources and properly mapping fields between profile sources to achieve the integration required is in the Discovery Server Control Center Guide.

**Authorization single sign-on**

After the user has authenticated to your Discovery Server, and the person profiles and spidered ACLs have been used to determine what a user sees in the K-map, what happens when the user tries to follow a link from within a search result or K-map? At this point, Discovery Server generates a URL or other link to the document in its original system, which is then opened in another browser window. It is then up to single sign-on solutions to ensure that the user is not prompted for authentication by their browser.

The specific single sign-on approaches that should be used vary based on the data repository type:

* **Domino, Quickplace, and Web/HTML sources**
  These data sources should be handled in the manner discussed in “Authentication single sign-on (SSO)” on page 214 since a regular HTTP URL is generated for these types or repositories.

* **Windows file system sources**
  The hand-off to Windows file system repositories is handled differently depending on the setup of your specific environment. Both the file URL (for example :\/) and http URL are passed down to the K-map front end. An ActiveX control used by the K-map UI determines if the file path expressed can be accessed by the client system. Normally, this is possible if the file path is within the Windows domain to which the client workstation is logged in. If the file reference works, then the file URL is used, and SSO is handled behind the scenes by normal Windows client and server domain authentication methods.

  If the file URL is not available, then the optional HTTP URL is used. However, this requires that your Windows NT/2000 environment has been set up to share the files via HTTP. If the HTTP URL works and is used, single sign-on should be handled as discussed in “Authentication single sign-on (SSO)” on page 214.

* **Microsoft Exchange sources**
  Exchange repositories are also handled differently depending on the setup of your specific environment. First, Discovery Server creates both a standard HTTP
URL and a Microsoft Outlook URL. The same ActiveX control that determines if the client system can access a file system is then used to parse this URL format and try to open the document using Microsoft Outlook. If that attempt fails, the K-map UI fails over to a standard browser HTTP request using the generated HTTP URL.

As with file system repositories, if the Microsoft Outlook reference is used, then SSO is handled by the Windows client and server infrastructure. Otherwise, SSO should be handled as discussed in “Authentication single sign-on (SSO)” on page 214.

### 8.2.3 Other directory integration considerations

One other directory integration aspect to consider for a Discovery Server deployment is in regard to the people awareness capabilities. For this to function properly, both your Discovery Server infrastructure and your Lotus Sametime infrastructure must be enabled for the same authoritative directory. This is required so that the user name you use to log in to Discovery Server is recognized as a valid user for your Sametime infrastructure.

**Tip:** If your Sametime 2.5 infrastructure is set up for LDAP directory usage, and not direct Domino directory usage, then the login name used to Discovery Server must be the LDAP Common name (that is, CN) because Sametime 2.5 only supports the LDAP common name for authentication purposes.

### 8.3 Data source integration via the XML spider

Discovery Server provides spiders for such sources as Domino databases, files in network file systems, and Web pages on Web servers. These sources need to be directly accessible to the Discovery server that is spidering them. In the case of Domino databases, this is controlled by Domino ACLs; for file systems, by the operating system access control; and, for Web systems, by Web security.

But what if you have a data source that cannot be specified using one of these spider definitions?

Some examples of such data sources are document management systems, ERP systems, and CRM systems. These systems have their own document and content control systems, and data is not readily available in a file or Web format to allow spidering, due to security or data storage reasons.

To address the need to spider such systems, Discovery Server includes an XML spider.
### 8.3.1 What is XML

XML (Extensible Markup Language) is a markup language similar to HTML (Hyper Text Markup Language). Whereas HTML is used to describe data and how it is to be displayed, XML defines data and its structure, but not its presentation. In other words, content and presentation are separate.

XML's strongest point is its ability to do data interchange. Because different organizations (or even different parts of the same organization) rarely standardize on a single set of tools, it takes a significant amount of work for two groups to communicate. XML makes it easy to send structured data across the Web so that nothing gets lost in translation.

When using XML, I can receive XML-tagged data from your system, and you can receive XML-tagged data from mine. Neither of us has to know how the other's system is organized. If another partner or supplier teams up with my organization, I don't have to write code to exchange data with their system. I simply require them to follow the document rules defined in the document type definition (DTD).

Let's illustrate this with an example.

**HTML**

```html
<p><r>Mr George Westwood<br>
123 Lotus Avenue<br>
Anytown 2000<br></p>
```

**XML**

```xml
<name>
<title>Mr</title>
<first-name>George</first-name>
<last-name>Westwood</last-name>
</name>
<address>
<street>12 Lotus avenue</street>
<city>Anytown</city>
<zipcode>2000</zipcode>
</address>
```

You can see that XML describes what the data is, whereas HTML just describes how it should look. Extensible Style Language (XSL) is used to describe how the XML content should be presented. The same XML information can be presented in different ways by using different XSL stylesheets.
8.3.2 How to use the XML spider

The XML spider is a general-purpose tool for spidering any data source. In this section we use a typical scenario of a document management system to illustrate how to use the XML spider.

The document management system uses a relational database to maintain metadata and document security; it uses the file system of its server to store the files (for example, MS Word documents, Lotus 1-2-3 files); and it provides a user interface (Web client or “thick” client).

In this scenario, it is unlikely that the spider will have access to the files managed by the document management system directly. If you were allowed to spider the file system for the document files, you would miss the metadata stored in the relational database.

What you need to do is assemble the document file and the metadata into a single source that can be spidered by the Discovery Server XML spider.

So, in this scenario, you write an extract program which connects to the document management system, generally using a published API from the document management system vendor. This extract program reads the relevant data and creates an XML file, as described later in this section. In addition to including any required metadata in the XML file, you can also reference the document file in the XML file, and the XML file will spider the document file as well as using the included metadata. In situations where it is not possible for the Discovery Server spider to access the document file directly in its original location, usually due to security policies, the extract program would also have to place a copy of the document file in a spider-accessible directory, as a temporary copy for the spider to use. Note that this temporary copy is only used by the spider and that access from the K-map will access the document file in its original location, with any required authentication.

Some things that you need to consider when creating the extract program are the following:

- Where do I store XML files and temporary document files?
- What security do I need for this temporary file area, that is, do I need to restrict access to only the Discovery server?
- Do I purge this temporary area periodically to minimize disk usage? (See “XML spider settings: Remove and Truncate” on page 230.)
– How do I handle document changes in the document management system, for example, do I record all draft changes or only version changes?
– How often do I want updates recognized in Discovery Server?

8.3.3 How the XML spider works

The XML spider works just like the other spiders. In the case of the XML spider, the data source is a directory in the file system of an accessible machine, which contains a set of XML files produced by some form of extract program. The XML spider consumes extracted/published XML, validates the XML against the Discovery Server DTDs, then uses XSL transformations to produce Internal XML to be processed by the K-map Building, Metrics, and K-map Indexing services.

Figure 8-3 illustrates the operations of the XML spider.

As you can see from Figure 8-3, the XML files to be spidered must exist in the file system. It is your responsibility to create the extract program to take data from the other repository and place it in the XML directory.
the data repository and create XML files on the file system that comply with the Discovery Server format requirements.

**XML file format requirements**

**DTD files**

In order for Discovery Server to spider XML data, the XML file must be constructed according to the DTDs provided for Discovery Server, which are stored on the Discovery server in the `\Lotus\DS\Data\discovery server xml directory`.

There are 3 DTDs for Discovery Server:

- server.dtd
- database.dtd
- document.dtd

The XML files can be reviewed in detail by simply viewing them in your favorite text editor. Additionally, usage details and definitions for the Discovery Server XML DTDs are included in the Discovery Server Control Center product documentation (ds_admin.chm).

These DTDs are used by the XML spider of Discovery Server to validate the format of the XML files to be spidered. Any errors in the XML files to be spidered will be reported in the spider log of Discovery Server.

**XML files**

You must specify a directory where the XML files are located; do this in the data source definition form of the Discovery Server Control Center. An example is shown in Figure 8-4.
There are at least 3 files that must exist in this directory for the XML spider to operate.

SERVER.XML  The server XML definition is required; its content must be formatted according to the server DTD.

DATABASE.XML  The database XML definition is required; its content must be formatted according to the database DTD.

DOCUMENT  The directory must contain at least one additional .XML file: the document XML definitions, formatted according to the document DTD. Each document .XML file represents information about a single document entity in the source data.
**Sample XML files**

This section provides a sample for each of the required .XML files, SERVER, DATABASE, and DOCUMENT. A table that follows each example gives further information about the tags in the example.

**SERVER.XML**

```xml
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE SERVER SYSTEM "\LDStraining\XMLdef\server.dtd">
<SERVER>
  <SERVERID>LDStraining</SERVERID>
</SERVER>
```

Table 8-1 gives an example of each tag in the SERVER.XML file. The Discovery Server product documentation provides additional information.

<table>
<thead>
<tr>
<th>Tag</th>
<th>Example</th>
</tr>
</thead>
</table>
| `<DOCTYPE SERVER SYSTEM>` | UNC format  
  `<DOCTYPE SERVER SYSTEM "\LDStraining\Lotus\DS\Data\discovery server xml\server.dtd">`
  Drive format  
  `<DOCTYPE SERVER SYSTEM "H:\Lotus\DS\Data\discovery server xml\server.dtd">`
  `where LDStraining is NT machine name and H:\ is the install drive for Discovery Server`
| `<SERVER>` | `<SERVERID>LDStraining</SERVERID>` |

**Attention:** The SERVER.XML must not be changed after the first XML spider processing of this XML source.

**DATABASE.XML**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE DATABASE SYSTEM "\LDStraining\Lotus\DS\Data\discovery server xml\database.dtd">
```
<DATABASE>
<DATABASEID>LOCALTEST</DATABASEID>
<NATIVEURL>\LDStraining\LOTUS\OS\DOMINO\HTML\XMLTESTDATA</NATIVEURL>
<HTTPURL>HTTP://LDStraining.lotus.com/XMLTESTDATA</HTTPURL>
<SUMMARY>Xml Test data for Spider</SUMMARY>
</ACL>
</DATABASE>

<table>
<thead>
<tr>
<th>Tag</th>
<th>Example</th>
</tr>
</thead>
</table>
| ```<DOCTYPE DATABASE SYSTEM>``` | UNC format  
```<DOCTYPE SERVER SYSTEM "\\LDStraining\Lotus\OS\Data\discovery server xml\database.dtd">```  
Drive format  
```<DOCTYPE SERVER SYSTEM "H:\Lotus\OS\Data\discovery server xml\database.dtd">```  
where LDStraining is NT machine name and H:\ is the install drive for Discovery Server |
| ```<DATABASE>``` |  
```<DATABASEID>DOCSET1</DATABASEID>``` |
| ```<NATIVEURL>``` | File drive URL  
```<NATIVEURL>file:///LDStraining/lotus/ds/data/domino/html/XMLTESTDATA/</NATIVEURL>``` |
| ```<HTTPURL>``` |  
```<HTTPURL>HTTP://LDStraining.lotus.com/XMLTESTDATA</HTTPURL>``` |
| ```<SUMMARY>``` |  
```<SUMMARY>This is a summary and is restricted to 256 characters</SUMMARY>``` |
| ```<OWNER>``` | Notes identity  
```<OWNER>Peter User/US/ACompany</OWNER>```  
File system identity  
```<OWNER>mydomain\peteru</OWNER>``` |
### Tag Example

#### Attention: The DATABASE.XML must not be changed after the first XML spider processing of this XML source.

**Document_N.XML**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE DOCUMENT SYSTEM "\\LDStraining\lotus\ds\discovery server xml\document.dtd">
<DOCUMENT>
  <IDENTIFIER>DOCUMENT2</IDENTIFIER>
  <AUTHOR>George User</AUTHOR>
  <UPDATEDBY>Paul User</UPDATEDBY>
  <CREATED>2002-05-25-08.00.00</CREATED>
  <LASTREAD>2002-05-27-08.00.00</LASTREAD>
  <MODIFIED>2002-05-30-08.10.00</MODIFIED>
  <REVISIONS>
    <REVISION>2002-05-25-10.00.00</REVISION>
    <REVISION>2002-05-26-08.00.00</REVISION>
    <REVISION>2002-05-29-08.10.00</REVISION>
  </REVISIONS>
  <TITLE>Document 2</TITLE>
  <BODY TYPE="TEXT"></BODY>
  <APPLICATION>MS Word</APPLICATION>
  <LANGUAGE>en</LANGUAGE>
</DOCUMENT>
```
Table 8-3 gives an example of each tag in the DOCUMENT.XML file. Read these examples in conjunction with the Discovery Server product documentation. The example assumes that the document files to be spidered are in the \lotus\ds\data\domino\html\xmltestdata directory. The .XML files can be in a directory different from the associated document files.

<table>
<thead>
<tr>
<th>Tag</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;!DOCTYPE DOCUMENT SYSTEM&gt;</td>
<td>UNC format</td>
</tr>
<tr>
<td></td>
<td>&lt;!DOCTYPE SERVER SYSTEM &quot;\LDStraining\Lotus\DS\Data\discovery server xml\document.dtd&quot;&gt;</td>
</tr>
<tr>
<td></td>
<td>Drive format</td>
</tr>
<tr>
<td></td>
<td>&lt;!DOCTYPE SERVER SYSTEM &quot;H:\Lotus\DS\Data\discovery server xml\document.dtd&quot;&gt;</td>
</tr>
<tr>
<td></td>
<td>where LDStraining is NT machine name and H:\ is the install drive for</td>
</tr>
<tr>
<td></td>
<td>Discovery Server</td>
</tr>
<tr>
<td>&lt;DOCUMENT&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;IDENTIFIER&gt;</td>
<td>&lt;IDENTIFIER&gt;Document number 1&lt;/IDENTIFIER&gt;</td>
</tr>
<tr>
<td></td>
<td>where identifier must be unique and must not change after first spider</td>
</tr>
<tr>
<td></td>
<td>runs.</td>
</tr>
<tr>
<td>&lt;DELETE&gt;</td>
<td>See Example 8-1</td>
</tr>
<tr>
<td>&lt;AUTHOR&gt;</td>
<td>&lt;AUTHOR&gt;George User&lt;/AUTHOR&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;AUTHOR&gt;Peter Name&lt;/AUTHOR&gt;</td>
</tr>
<tr>
<td>&lt;UPDATEDBY&gt;</td>
<td>&lt;UPDATEDBY&gt;George User&lt;/UPDATEDBY&gt;</td>
</tr>
<tr>
<td>&lt;CREATED&gt;</td>
<td>&lt;CREATED&gt;2002.06.23-08.00.12&lt;/CREATED&gt;</td>
</tr>
<tr>
<td></td>
<td>format is YYYY-MM-DD-HH.MM.SS</td>
</tr>
<tr>
<td>&lt;LASTREAD&gt;</td>
<td>&lt;LASTREAD&gt;2002.06.23-08.00.12&lt;/LASTREAD&gt;</td>
</tr>
<tr>
<td></td>
<td>format is YYYY-MM-DD-HH.MM.SS</td>
</tr>
<tr>
<td>Tag</td>
<td>Example</td>
</tr>
<tr>
<td>----------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>&lt;MODIFIED&gt;</code></td>
<td><code>&lt;MODIFIED&gt;2002.06.23-08.00.12&lt;/MODIFIED&gt;</code></td>
</tr>
<tr>
<td></td>
<td>format is YYYY-MM-DD-HH.MM.SS</td>
</tr>
<tr>
<td><code>&lt;REVISIONS&gt;</code></td>
<td><code>&lt;REVISIONS&gt;</code></td>
</tr>
<tr>
<td>AND <code>&lt;REVISION&gt;</code></td>
<td><code>&lt;REVISION&gt;2002.06.25-08.00.12&lt;/REVISION&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;REVISION&gt;2002.06.28-10.00.12&lt;/REVISION&gt;</code></td>
</tr>
<tr>
<td></td>
<td>format is YYYY-MM-DD-HH.MM.SS</td>
</tr>
<tr>
<td><code>&lt;FIELD&gt;</code></td>
<td>Reserved tag, do not use</td>
</tr>
<tr>
<td><code>&lt;SUMMARY&gt;</code></td>
<td><code>&lt;SUMMARY&gt; This is a summary and is restricted to 256 characters&lt;/SUMMARY&gt;</code></td>
</tr>
<tr>
<td><code>&lt;KEYWORDS&gt;</code></td>
<td><code>&lt;KEYWORDS&gt;</code></td>
</tr>
<tr>
<td>and <code>&lt;KEYWORD&gt;</code></td>
<td><code>&lt;KEYWORD&gt;TERM1&lt;/KEYWORD&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;KEYWORD&gt;TERM2&lt;/KEYWORD&gt;</code></td>
</tr>
<tr>
<td><code>&lt;TITLE&gt;</code></td>
<td><code>&lt;TITLE&gt;My document title&lt;/TITLE&gt;</code></td>
</tr>
<tr>
<td><code>&lt;SUBJECT&gt;</code></td>
<td><code>&lt;SUBJECT&gt; This is the subject of this document and is restricted to 256 characters&lt;/SUBJECT&gt;</code></td>
</tr>
<tr>
<td><code>&lt;BODY&gt;</code></td>
<td><code>&lt;BODY&gt;This is the body of the document and will be analysed for categorization along with any INCLUDED file&lt;/BODY&gt;</code></td>
</tr>
<tr>
<td></td>
<td>See also INCLUDE tag</td>
</tr>
<tr>
<td><code>&lt;APPLICATION&gt;</code></td>
<td><code>&lt;APPLICATION&gt;Lotus Wordpro&lt;/APPLICATION&gt;</code></td>
</tr>
<tr>
<td><code>&lt;LANGUAGE&gt;</code></td>
<td><code>&lt;LANGUAGE&gt;en-US&lt;/LANGUAGE&gt;</code></td>
</tr>
<tr>
<td></td>
<td>or</td>
</tr>
<tr>
<td></td>
<td><code>&lt;LANGUAGE&gt;en&lt;/LANGUAGE&gt;</code></td>
</tr>
<tr>
<td></td>
<td>Note that language tag is case-sensitive</td>
</tr>
<tr>
<td><code>&lt;USENOTES&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;NATIVEURL&gt;</code></td>
<td>File drive URL</td>
</tr>
<tr>
<td></td>
<td><code>&lt;NATIVEURL&gt;file:///LDStraining/lotus/ds/data/domino/html/XMLTESTDATA/Sample document.doc&lt;/NATIVEURL&gt;</code></td>
</tr>
<tr>
<td><code>&lt;HTTPURL&gt;</code></td>
<td><code>&lt;HTTPURL&gt;HTTP://LDStraining.lotus.com/XMLTESTDATA/Sample document.doc&lt;/HTTPURL&gt;</code></td>
</tr>
<tr>
<td>Tag</td>
<td>Example</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| `<ACL>`      | `<ACL DOMAINTYPE="windowsNT" DEFAULTDOMAIN=""/>
`<DENY>thisusergroup</DENY>`<br>`<DENY>thisuser</DENY>`<br>`<ALLOW>thisusergroup1</ALLOW>`<br>`</ACL>`<br>`See DATABASE.XML example for Domino ACL example`<br>`NOTE: If you wish to use the `<ACL>` tag in the DOCUMENT.XML file, you must create an `<ACL>` tag in the DATABASE.XML, with at least one `<ALLOW>` or `<DENY>` tag. This tag can be empty if you want all users to access the database, and to then control access at a document level. If you do not include this in the DATABASE.XML, you will receive an ACL error when spidering DOCUMENT.XML. |
| `<INCLUDE FILEPATH>` | **UNC format**<br>`<INCLUDE FILEPATH="\LDStraining\lotus\ds\data\domino\html\Sample Document.DOC"></INCLUDE>`<br>`Drive format`<br>`<INCLUDE FILEPATH="H:\lotus\ds\data\domino\html\Sample Document.DOC"></INCLUDE>` |
| `<ATTACHMENT>` | `<ATTACHMENT FILEPATH="\LDStraining\lotus\ds\data\domino\html\xmltestdata\Sample Document1.DOC" IDENTIFIER="attachment1" NATIVEURL="file:///LDStraining/LOTUS/DS/DOMINO/HTML/XMLTESTDATA/Sample document1.doc" HTTPURL="HTTP://LDStraining.lotus.com/XMLTESTDATA/Sample document1.doc" USENOTES="0"/>`
<br>`<ATTACHMENT FILEPATH="\LDStraining\lotus\ds\data\domino\html\xmltestdata\Sample Document2.DOC" IDENTIFIER="attachment2" NATIVEURL="file:///LDStraining/LOTUS/DS/DOMINO/HTML/XMLTESTDATA/Sample document2.doc" HTTPURL="HTTP://LDStraining.lotus.com/XMLTESTDATA/Sample document2.doc" USENOTES="0"/>` |
Delete Document.XML

**Note:** This XML file format allows deletion of an existing document from the Discovery Server.

**Example 8-1**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE DOCUMENT SYSTEM "\\LDStraining\Lotus\DS\discovery server xml\document.dtd">
<DOCUMENT>
  <IDENTIFIER>DOCUMENT2</IDENTIFIER>
  <DELETE/>
</DOCUMENT>
```

The IDENTIFIER tag must contain a value matching an existing document that Discovery Server processed in a previous spider run.

**XML Spider operation**

When the XML spider runs on a defined data source, which files are processed is dependent on the “Process all documents during next run” setting on the spider definition form.

If this setting is not selected, it checks the specified directory for new .XML files. It does this by checking for non-zero byte .XML files in the directory (see “XML spider settings: Remove and Truncate” on page 230). For any non-zero byte .XML files, the spider examines the <IDENTIFIER> tag and the <MODIFIED> tag. If the <IDENTIFIER> tag has a value the spider has not processed before, this is a new document to be included in the K-map. If the <IDENTIFIER> tag has a value the spider has processed before (that is, an existing document in the K-map), the <MODIFIED> tag is checked for a new date and time. A new date and time means that the document needs to be updated in Discovery Server.

If the spider finds no change in the file, then no action is taken.

**Tip:** It is important to note this, because, during your testing of the XML spider on one of your data sources, you might have updated a file in the file system, but nothing in the .XML file has changed. The XML spider will not pick up this file and process it unless you have specified “Process all documents in the next run” in the data source definition. If you don’t want to process all files every time, the easiest way to “force” a change is to modify the <MODIFIED> tag with a new date and/or time.
**XML spider settings: Remove and Truncate**

As you will have realized from the description of the operation of the XML spider, the XML spider, in most instances, works on a copy of the data, rather than the source data directly.

This means that there will be data duplication, which could have storage implications for large data sources.

The Discovery Server XML spider provide two options you can use to control the amount of storage used by the .XML files. Both options are available on the data source definition form for the XML spider in the Discovery Server Control Center. This setting is set for each XML spider source.

The “After processing” field has 3 options:

- **Leave XML source files unchanged**
  
  This option leaves the XML files as they are. Your extract program will need to overwrite or update the relevant files when data changes in the source.

- **Truncate XML source files**
  
  This option truncates the .XML fields to zero bytes, but leave them in the directory. This option is useful if an administrator wishes to track which files have been processed. It also improves throughput for subsequent spider runs, as zero byte files will be ignored by the spider.

- **Remove XML source files**
  
  This option deletes the XML source files. This option provide the greatest space savings. The XML spider does not delete any files in the file system that have been INCLUDED or attached via an ATTACHMENT tag in the XML files.

**Conclusion**

The XML spider gives you the ability to spider data from any source appropriate to your needs. In most cases you will need to write an extract program, but this a small overhead for making all your information sources available in your Discovery Server environment.
8.4 Integration using Discovery Server programming interfaces

8.4.1 The Lotus Discovery Server API Toolkit

API overview and purpose
The Lotus Discovery Server API Toolkit (Discovery Server API) provides a set of Java classes that encapsulate many key services of Lotus Discovery Server, such as search, metrics calculations, K-map taxonomy content, and expertise profiles. This toolkit enables developers to build applications for the Lotus Discovery Server, and to integrate Discovery Server with existing CRM, ERP, SCM, or portal infrastructure solutions.

These Java classes provide a way to:
- Build alternative user interfaces to Discovery Server, for example, to replace the K-map, hotlist windows, or category monitors
- Build user interfaces that support other browsers (the out-of-the-box Discovery Server K-map only supports Internet Explorer)
- Extract information that Discovery Server collects (for example, to build customized metrics reports)
- Query or update taxonomy information, including importing existing file-based taxonomy and content
- Retrieve profile information maintained by Discovery Server, such as affinities
- Build automated search queries

You can use the Discovery Server API to develop applets, servlets, and standalone Java applications.

API samples to get you started
The API includes the source code for many sample applications. These samples need to be compiled using a Java compiler and installed in your Discovery server before they can be used. All of the samples are provided as guidelines only, and should be carefully analyzed and hardened by a Java developer prior to production usage.

The following sample applications are provided within the toolkit:
- Metrics servlet** - query metrics information maintained by Discovery Server
- Cursor applet and application** - a Discovery Server category browser
Getting started with the toolkit

The Discovery Server API is available on the product CD in the extras\KDSapi directory. The Discovery Server API requires a compatible Java Virtual Machine (JVM); the install program will search for a compatible JVM on your PC. If none is found, the toolkit will install a 1.1.8 Java Runtime Environment along with the toolkit in a toolkit subfolder named \_jvm. This ensures that the toolkit has a compatible JVM to run the uninstall program. The Discovery Server API should be installed on a client machine, not a server machine. The default install location is C:\KDSAPI.

Be sure to look at the API documentation, KDSAPIHELP.CHM, available when you install the Discovery Server API. This documentation is very thorough, and will allow any experience Java developer to quickly come up to speed with the toolkit's capabilities and usage. Additional documentation for some of the samples mentioned is included within the specific samples directory, below the KDS API directory.

Attention: Those samples marked with **, delivered with Discovery Server 2.0, have not been updated from samples supplied with Discovery Server 1.1. They do not take account of new features in the API in 2.0. Take the samples as a starting point and use the documentation to add features that you require in your programs.

As an example, Discovery Server 2.0 now provides a hidden category capability for category not yet published. The samples included in the toolkit use API calls that return the entire category set, and you need to parse this to determine which categories are hidden, if you do not want to show all categories.

- K-map browser servlet** - HTML-only format for K-map, allowing support for a wider range of browsers
- K-map Search servlet** - HTML-only K-map search
- K-map pooling servlet** - provides connection pooling for better performance for search and browse servlets
- Taxonomy Import - shows how to import data from a repository into the taxonomy tree (see “New tools for your taxonomy” on page 181)
- Rules-based classifier - shows how to classify documents from repositories into the taxonomy tree
- Workqueue application - shows how to place a user-defined WorkQueue between existing WorkQueues.
8.4.2 An alternative API - XML output from the UI servlets

Discovery Server uses servlets to deliver XML data that is formatted for display of the K-map in the browser. The XML is produced as a result of calls to the Discovery Server servlets.

As an alternative to using the Discovery Server API to build a custom user interface to the K-map via custom servlets, applets, and so forth, developers may choose to use the direct XML output from the out-of-the-box K-map servlet. If you have strong skills in XML or other languages such as PERL, but do not have strong Java skills, working with the XML output may be easier. Alternatively, should you simply require the same functionality as the existing KMAP interface, but desire a different look or feel, then again the XML output may be the easier path.

Calling the K-map servlet

To use the servlet to return XML, you simply call the servlet directly, providing URL parameters when calling the servlet.

The original K-map URL which resulted in HTML:

http://itsolds1.lotus.com/kmap.htm

becomes the following to receive just the XML in return:

http://itoslds1.lotus.com/servlet/KmapServlet

and you must then add parameters, just as parameters are often included in calls to the K-map interface.

So, an original K-map URL which resulted in HTML:

http://itsolds1.lotus.com/kmapbrowse.htm?path=0000000012345678&cmd=b

becomes the following to receive just the XML in return:

http://itsolds1.lotus.com/servlet/KmapServlet?path=0000000012345678&cmd=b

Note: KmapServlet is case-sensitive.

There are parameters for paging, searching, launching from a portal, and more; but there are some minimum parameters you need to be aware of:

► A call to the home page requires no parameters.
► When browsing to a category, the minimum parameters are path=\<the category id\> and cmd=\.

Chapter 8. Integrating Discovery Server into your enterprise  233
When searching, the minimum parameters are scope=<the search scope>, search=<the search text>, refine=<the refinement we are on (0 for the original search)>, and cmd=s.

Table 8-4 and Table 8-5 provide details on the parameters that are used with the KmapServlet. Example 8-2 then gives an example of the XML returned from a call to the KmapServlet.

### Table 8-4  Common parameters

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Data type</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>path</td>
<td>string</td>
<td>&quot;16 character numeric string&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PATH is a cluster ID from the content map that navigates the taxonomy to the last cluster. KMap uses PATH to bookmark categories. PATH will also be used to search within a category when available. Example: path=0987654321654321</td>
</tr>
<tr>
<td>cmd</td>
<td>string</td>
<td>Indicates the operation to be performed on the server. Values are as follows: b browse s search rlocx return client locale strings as XML err returns test error XML content</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Example: cmd=b</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Note: Other CMD options are available. However, they are not included here since they do not return XML.</td>
</tr>
<tr>
<td>tz_offset</td>
<td>string</td>
<td>The time zone offset from GMT measured in minutes. It is subtracted from the UTC time stored in the database. This value is retrieved by calling the getTimezoneOffset() method in the JavaScript Date object. See the Dates section for more details. Example: tz_offset=240</td>
</tr>
<tr>
<td>dpg</td>
<td>string</td>
<td>The desired page number for document results. Example: dpg=1</td>
</tr>
<tr>
<td>ppg</td>
<td>string</td>
<td>The desired page number for people results. Example: ppg=1</td>
</tr>
<tr>
<td>rcpg</td>
<td>string</td>
<td>The desired page number for related category results. Search only. This is not implemented in Discovery Server 2.0.</td>
</tr>
</tbody>
</table>
Table 8-5  Search-specific parameters

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Data type</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>docsrt</td>
<td>string</td>
<td>Document table sort field. Value is used to determine how the table should be sorted. Sort on one of the following:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>None 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Creation Date 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Modification Date 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Document Type 3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Author 4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Value 5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Fit 6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DocChange 7</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Source 8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Title 9</td>
</tr>
<tr>
<td>docsrt</td>
<td></td>
<td>Example: docsrt=0</td>
</tr>
<tr>
<td>docsrtdir</td>
<td>string</td>
<td>Document table sort direction. 0 indicates default sort order and 1 indicates reverse sort order.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Example: docsrtdir=1</td>
</tr>
<tr>
<td>pplsrt</td>
<td>string</td>
<td>People table sort field. Value is used to determine how the table should be sorted. Sort on one of the following:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Name 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Affinity 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Job Title 3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Location 4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Dept 5</td>
</tr>
<tr>
<td>pplsrt</td>
<td></td>
<td>Example: pplsrt=5</td>
</tr>
<tr>
<td>pplsrt</td>
<td></td>
<td>People table sort direction. 0 indicates default sort order and 1 indicates reverse sort order.</td>
</tr>
<tr>
<td>pplsrt</td>
<td></td>
<td>Example: pplsrt=0</td>
</tr>
<tr>
<td>search</td>
<td>non-blank string</td>
<td>search is a non blank string that is the query to search for. A search string with multiple tokens should have the space characters converted to the '+=' character. The '+' character is not added as an operator to the query. It is used to separate tokens of the search string. Multiple values can be passed, separated by the carot (^) symbol. Example: search=knowledge management</td>
</tr>
</tbody>
</table>
Example 8-2 shows the XML returned by the servlet when browsing to a category.

The following URL with parameters was used to create this XML response:
http://itsolds1.lotus.com/servlet/KmapServlet?sw=0&path=85256B370070BA85&cmd=b

Returned is global information (list counts), the navigation history, which gives the category title and its id for each category in the navigation history (which appears at the top of the UI lists), and document table and people table information. Not every piece of data you see in the XML is displayed in the out-of-the box UI, so you will need to determine what data your would like to use

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Data type</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>refine</td>
<td>string &quot;0..n&quot;</td>
<td>refine represents the number of refinements that have been made. Example: refine=0</td>
</tr>
<tr>
<td>searchType</td>
<td>string</td>
<td>searchType represents the type of data to be returned by a query. One of the following: Everything -1 Document 0 People 1 Related Categories 2 Places 3 Example: searchType=-1</td>
</tr>
<tr>
<td>scope</td>
<td>string, &quot;0-7&quot;</td>
<td>scope is a string value ranging from 0 through 7 that determines query scope. It directly correlates to the scope choices in the KMap UI. 0 = everything about... 7 = categories about. Multiple values can be passed, separated by the carot (^) symbol. When refine is greater than 0, scope could have multiple values like scope=&quot;0^2&quot;&amp;refine=1. Used along with refine. The scope order in KMap list box: everything about [default] 0 documents about 1 documents authored by 2 people named 3 people who know about 4 people whose Profile contains 5 places about 6 categories about 7 Example: scope=0^3</td>
</tr>
<tr>
<td>sppg</td>
<td>string</td>
<td>The desired page number for document results. Example: sppg=1</td>
</tr>
</tbody>
</table>

Example 8-2 shows the XML returned by the servlet when browsing to a category.
in your UI. Some of the XML is used by the K-map UI to format links so that when you click on a link you have the information you need to do the action desired.

Example 8-2 XML example

```xml
<xml version="1.0" encoding="UTF-8" >
<KmapEntity>
    <global>
        <doc_total>2</doc_total>
        <people_total>1</people_total>
        <place_total>0</place_total>
        <relcat_total>0</relcat_total>
    </global>
    <nav_history>
        <level>
            <title>Home</title>
            <cid>000000012345678</cid>
        </level>
        <level>
            <title>hunger care oxfam</title>
            <cid>85256B3700708A85</cid>
        </level>
    </nav_history>
    <categoryTable />
    <docTable>
        <document>
            <title>German - About Hunger FAKTEN ÜBER DEN HUNGER</title>
            <summary>The Hunger Site - Translations FAKTEN ÜBER DEN HUNGER 1. Jeden Tag sterben ca. 24.000 Menschen an Hunger oder durch hungerbezogene Ursachen. Die Zahl der Sterbenden ist gesunken, von 35.000 vor zehn Jahren und von 41.000 vor zwanzig Jahren.</summary>
            <url>HTTP://sneaker.iris.com/LDSsmoke.nsf/0/4CFD66B7849DB9F085256B6D0053FA3C%3FOpen</url>
            <native_url>NOTES:///85256B1000734350/0/4CFD66B7849DB9F085256B6D0053FA3C%3FOpen</native_url>
            <value>0</value>
            <author>Dane Johnson/Iris</author>
            <date_created>1/21/00</date_created>
            <date_modified>11/26/01</date_modified>
            <source>LDSSmoke database</source>
            <source_url>http://sneaker.iris.com/LDSSmoke.nsf</source_url>
            <source_nativeURL>notes://sneaker.iris.com/LDSSmoke.nsf</source_nativeURL>
            <doctype>Lotus Notes</doctype>
            <use_notes>1</use_notes>
            <docid>85256B37006FEE18</docid>
        </document>
    </docTable>
</KmapEntity>
```
<summary>The Hunger Site - Translations DE FEITEN OVER HONGER 1. Elke dag sterven er zo&deg;39;n 24.000 mensen van honger of aan de gevolgen ervan.</summary>
<url>HTTP://sneaker.iris.com/LDSsmoke.nsf/0/A95707D17DFA0F358525686D00541D0E%3FOpen</url>
<native_url>NOTES:///85256B1000734350/0/A95707D17DFA0F358525686D00541D0E%3FOpen</native_url>
<value>0</value>
<author>Dane Johnson/Iris</author>
<date_created>1/21/00</date_created>
<date_modified>11/26/01</date_modified>
<source>LDSsmoke database</source>
<source>http://sneaker.iris.com/LDSsmoke.nsf</source>
<source_nativeURL>notes://sneaker.iris.com/LDSsmoke.nsf</source_nativeURL>
<doctype>Lotus Notes</doctype>
<use_notes>1</use_notes>
<docid>85256B37006FEFE7</docid>

Data returned by K-map servlet
This section describes the XML data that is returned from calls to the K-map servlet.
**Global data**
The global tag will contain information that does not fit in any other object. This information includes the following:

- Total number of document in the current category filtered according to the user's access
- Total number of people in the current category
- Total number of places in the current category

**Example 8-3  Global tag example**

```xml
<global>
  <doc_total>10</doc_total>
  <people_total>3</people_total>
  <place_total>0</place_total>
  <relcat_total>0</relcat_total>
  <PlaceBrowseEnabled>0</PlaceBrowseEnabled>
  <PlaceSearchEnabled>0</PlaceSearchEnabled>
</global>
```

**Navigation history**
The navigation history will be passed down to the browser as XML along with other category-related information (documents, people, places, and so forth).

The container for this data will be called `nav_history` and it will contain categories in order from top to bottom. Each category will have a tag of `level` and will be a subset of the `categoryTable` category object. The only attributes that it will have are `cid` and `title`.

**Example 8-4  Navigation history**

```xml
<nav_history>
  <level>
    <title>Home</title>
    <cid>0000000012345678</cid>
  </level>
  <level>
    <title>filename command character</title>
    <cid>85256A4000800199</cid>
  </level>
  <level>
    <title>directory iris</title>
    <cid>85256A400080019E</cid>
  </level>
</nav_history>
```
**Categories**
For categories, the servlet returns the XML for display of categories.

The name of the tag that is the container of categories is `categoryTable`. There are two types of categories in the `categoryTable`, category and symbolic link. The tag for both categories and symbolic links is `category`. Links and categories are differentiated by the type property tag (`regular` for categories and `symbolic` for symbolic links). The property tags for the category objects are shown in Table 8-6.

**Table 8-6  Category properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cid</td>
<td>The category's cluster ID</td>
</tr>
<tr>
<td>title</td>
<td>The category's label</td>
</tr>
<tr>
<td>type</td>
<td><em>regular</em> if it is a category; <em>symbolic</em> if it is a symbolic link</td>
</tr>
<tr>
<td>subcat</td>
<td>Number of sub-categories within the category</td>
</tr>
<tr>
<td>value</td>
<td>Search only; fulltext relevance score</td>
</tr>
</tbody>
</table>

**Example 8-5  Category example**
```xml
<?xml version="1.0" encoding="UTF-8" ?>
<categoryTable>
  <category>
    <cid>85256A4000800199</cid>
    <title>filename command character</title>
    <type>regular</type>
    <subcat>3</subcat>
  </category>
  <category>
    <cid>85256A400080019B</cid>
    <title>install</title>
    <type>regular</type>
    <subcat>3</subcat>
  </category>
  <category>
    <cid>85256A4000800110</cid>
    <title>knowledge management business</title>
    <type>regular</type>
    <subcat>2</subcat>
  </category>
  <category>
    <cid>85256A400080019A</cid>
    <title>pointer stack block</title>
```
Documents
For documents, a servlet returns document tables to display documents in browse and search.

The name of the tag that is the container of documents is `docTable`. The tag for individual documents is `document`. The property tags for each document object are shown in Table 8-7.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>title</td>
<td>Document's title</td>
</tr>
<tr>
<td>summary</td>
<td>Document summary</td>
</tr>
<tr>
<td>url</td>
<td>The HTTP URL of the document</td>
</tr>
<tr>
<td>native_url</td>
<td>Optional; file URL or Notes URL of document</td>
</tr>
<tr>
<td>value</td>
<td>Document's affinity score in browse and fulltext relevance score in search</td>
</tr>
<tr>
<td>author</td>
<td>Document's author</td>
</tr>
<tr>
<td>date_created</td>
<td>Document's creation date</td>
</tr>
<tr>
<td>date_modified</td>
<td>Date that the doc was last modified</td>
</tr>
<tr>
<td>source</td>
<td>The repository that document comes from</td>
</tr>
<tr>
<td>source_url</td>
<td>HTTP URL to the repository</td>
</tr>
<tr>
<td>source_nativeURL</td>
<td>Native URL to the repository, if applicable</td>
</tr>
<tr>
<td>doctype</td>
<td>Document's type (Word, HTML file, etc)</td>
</tr>
<tr>
<td>use_notes</td>
<td>Whether or not Notes is the preferred viewer; 1 to use Notes, 0 to not</td>
</tr>
</tbody>
</table>
Example 8-6  Document example

<?xml version="1.0" encoding="UTF-8" ?>
<docTable>
  <document>
    <title>Attachment - ASCII Text International</title>
    <summary>Document CN=Dane Johnson/O=Iris;Anonymous</summary>
    <url>HTTP://sneaker.iris.com/LDSsmoke.nsf/0/CB635C5F478E8A7E85256984006DD1CE?Open</url>
    <native_url>NOTES:///85256B1000734350/0/CB635C5F478E8A7E85256984006DD1CE?Open</native_url>
    <value>0</value>
    <author>Dane Johnson/Iris</author>
    <date_created>26.10.2000</date_created>
    <date_modified>26.11.2001</date_modified>
    <source>Discovery Server Smoke</source>
    <source_nativeURL>notes://sneaker.iris.com/LDSsmoke.nsf</source_nativeURL>
    <doctype>Lotus Notes</doctype>
    <use_notes>1</use_notes>
    <docid>85256B6F0053F7F2</docid>
    <altNameAuthor />
  </document>
</docTable>

People
For people, the servlet returns a people table to present people in browse and search.

The name of the tag that is the container of people is peopleTable. The tag for individual documents is person. The property tags for each person object are shown in Table 8-8.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>docid</td>
<td>The document ID</td>
</tr>
<tr>
<td>altNameAuthor</td>
<td>The author's alternate name (used for display)</td>
</tr>
</tbody>
</table>
### Table 8-8  People properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>abbreviatedName</td>
<td>Person's abbreviated canonical name (for example, John Doe/Lotus)</td>
</tr>
<tr>
<td>niceName</td>
<td>Person's name without domain</td>
</tr>
<tr>
<td>url</td>
<td>URL of the person's profile</td>
</tr>
<tr>
<td>phone_no</td>
<td>Person's phone number</td>
</tr>
<tr>
<td>value</td>
<td>Person's affinity to the current category in browse and fulltext relevance score in search</td>
</tr>
<tr>
<td>jobTitle</td>
<td>Person's job title</td>
</tr>
<tr>
<td>location</td>
<td>Person's location (e.g. Westford)</td>
</tr>
<tr>
<td>department</td>
<td>Department that a person works in</td>
</tr>
<tr>
<td>email</td>
<td>Person's e-mail address</td>
</tr>
<tr>
<td>affinity1</td>
<td>Pipe delimited affinity data; namefull path</td>
</tr>
<tr>
<td>affinity2</td>
<td>Same as affinity1</td>
</tr>
<tr>
<td>affinity3</td>
<td>Same as affinity1</td>
</tr>
<tr>
<td>affinity4</td>
<td>Same as affinity1</td>
</tr>
<tr>
<td>affinity5</td>
<td>Same as affinity1</td>
</tr>
<tr>
<td>more_affinities</td>
<td>Indicates whether or not there are additional affinities. 1 means there are, 0 indicates there are not</td>
</tr>
<tr>
<td>altAbbreviatedName</td>
<td>Person's alternate name</td>
</tr>
</tbody>
</table>

### Example 8-7  People example

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<peopleTable>
  <person>
    <abbreviatedName>Carol Easthope/ITSO</abbreviatedName>
    <niceName>Carol Easthope</niceName>
    <url>/people.nsf/(lookupview)/Carol Easthope</url>
    <phone_no>555-1111</phone_no>
    <value>3</value>
    <jobTitle>Software Engineer</jobTitle>
    <location>Someplace</location>
    <department>Knowledge Management</department>
  </person>
</peopleTable>
```
Metrics tracking
If you wish to provide metrics tracking via the user interface you build when using servlet XML output, you will need to make calls to the servlet to record the interaction in the user interface.

Metrics tracking is done as the user interacts with the K-map. Metrics are tracked as the user opens up a document or person.

The client UI will make a servlet call using the appropriate **cmd** parameter for the type of tracking to be performed. For tracking from the browse UI, the document ID or person common name will be passed up in the **id** parameter. For tracking from the search UI, an additional **search** parameter with the search text will be passed to the servlet. The servlet tracks a gotocategory action from the servlet, and an additional **cmd2** parameter is passed up with the search text if the gotocategory command was initiated from the search page (via a “go to related category” link).

8.5 Complementary products
While Discovery Server provides significant value through search, profiling, affinity mapping, and taxonomy browsing, there are a number of complementary products that deliver additional functionality to Discovery Server. This section discusses some of these products.

8.5.1 IBM WebSphere Portal Family
The WebSphere Portal Family provides a suite of technologies that allow you to build rich, dynamic, and collaborative intranet and Internet portals for employees and customers.
How WebSphere Portal complements Discovery Server
As mentioned previously, the Discovery Server user interface is likely to be delivered within an existing portal framework. IBM’s portal offering, the Websphere Portal Family, delivers several key pieces of integration with the Discovery Server:

Find/Search Integration
Websphere Portal Family 4.1, in the “Extend” and “Experience” versions, includes a Find button which can be defined to open the Discovery Server K-map for all portal searches. Figure 8-5 and Figure 8-6 show the find button as it appears on the standard portal theme, as well as the interface for enabling this integration.

![Figure 8-5 The default Websphere Portal find button](image-url)
Discovery Server portlets
The Discovery Server development team is working on additional integration with Websphere Portal Family, including K-map browser portlets, Discovery Server search portlets, and profile portlets. As these capabilities are made available they will be announced, and made available for download, via the Lotus Developer Domain website.

Discovery Server Collaborative Components
The WebSphere Portal Family 4.1, in the “Extend” and “Experience” versions, also includes some simplified API wrappers to enable easier creation of basic portlets that include Discovery Server capabilities. These Discovery Server API services are available as part of the Portal Extend Collaborative Components API.

It is important to note that the Discovery Server services included with WebSphere Portal Extend v4.1 were layered over the Discovery Server 1.1 release API. As the API in the previous release of Discovery Server did not included some of the performance enhancements in the 2.0 release API, such as
connection pooling, the performance impact of utilizing these components in your portal server environment should be fully tested.

The simplified DiscoveryServerService methods provided in this Collaborative Components API fall into the following categories:

- Search APIs
- APIs which return URLs for use in UI
- APIs which return core Discovery Server objects from a Discovery Server

Following are some examples of common search queries that can be performed using this DiscoveryServerService Collaborative Components API:

- Searching for everything about a subject:
  To search for "knowledge management", pass "knowledge management OR knowledge* AND management*" as the query to searchDocuments, searchPeople, searchCategories, and searchPlaces (if places are installed and configured with K-station).

- Searching for documents about a subject:
  To search for "knowledge management", pass "knowledge management OR knowledge* AND management*" as the query to searchDocuments.

- Searching for documents authored by a specific person:
  To search for "John Doe", pass "[$UpdatedBy] CONTAINS John Doe" as the query to searchDocuments.

- Searching for people by name:
  To search for "John Doe", pass "[fullname] CONTAINS John Doe" as the query to searchPeople.

- Searching for people who know about a specific subject:
  To search for "knowledge management", pass "[AffinityPublishedLeaf] CONTAINS knowledge management OR knowledge* AND management*" as the query to searchPeople.

- Searching for people whose Profile contains a specific piece of information:
  To search for "knowledge management", pass "knowledge management OR knowledge* AND management*" as the query to searchPeople.

- Searching Categories for specific information:
  To search for "knowledge management", pass "knowledge management OR knowledge* AND management*" as the query to searchCategories.

- Searching Places for specific information (when installed and configured with K-station):
To search for "knowledge management", pass "knowledge management OR knowledge* AND management*" as the query to searchPlaces. People related information (almost all methods use HTTP)

For more information, see:

www.ibm.com/websphere/portalfamily

8.5.2 IBM Lotus Extended Search

Extended Search provides parallel, distributed, heterogeneous search capability across Notes domains, legacy data stores, and the Internet, all from within the Notes environment or via a Web browser.

The result is single-point access to a variety of data stores, including Domino 4.5 and higher, relational databases (IBM DB2, Oracle, Sybase, MS SQL Server, MS Access), Domino.Doc, MS Index Server, MS Site Server, MS Exchange, and over 18 Web search sites. Extended Search allows you to specify a single search query, and retrieve results from all or any of these sources into a single search results screen.

With Extended Search, enterprises can streamline access to internal and external information, improve company and partner communications, and increase the effectiveness and scope of reference of their existing knowledge workers.

How Extended Search complements Discovery Server

Extended Search uses connectors to access data to be included in a search query.

An Extended Search connector for Discovery Server is under development as of the publication date of this book. This connector, when available, will allow Extended Search to include search results from Discovery Server in the results set, along with result sets from other data sources searched in the same search query.

Extended Search is useful when you want to combine search results from a number of sources with search results from Discovery Server. It is unlikely that you would include every data source in your enterprise within your Discovery Server deployment, for reasons outlined in 6.3.1, “Choosing repositories” on page 161. Also, if you decide on a Discovery Server architecture that includes more than one Discovery Server deployment in your organization, Extended Search gives you a way of searching across multiple independent Discovery Server environments. Figure 8-7 on page 249 shows an example of such a federated search configuration that includes Discovery Server.
8.5.3 IBM Lotus Sametime

Efficient, flexible, instantaneous communication is critical to a company's success. Sametime supports immediate communication with people across the hall or around the world, either through secure text messaging, audio and video, or full collaborative meetings.

The Sametime family includes the Sametime server, the Sametime Connect client, and a range of Application Developer tools. The Sametime Server also works seamlessly with any browser or with Lotus Notes, and has audio and video capabilities to enhance your online experience.

The three foundations of real-time collaboration are awareness, conversation, and shared objects with audio and video.

For more information, see:
www.lotus.com/extendedsearch
Awareness means a member of a team is aware of when other members are online. You can share your awareness through Sametime Connect, a Web page, or through a mobile device.

Conversations with others can easily be started using instant text messages or a chat session involving many people. With Sametime 2.0 you can also have audio/video sessions over IP.

Shared Object capabilities enable team members to share live documents or applications with others. For example, you can let others control your screen so you can collaborate on project regardless of geographic distance.

How Sametime complements Discovery Server
Discovery Server is Sametime-enabled and installation of a Sametime server will allow you to make the real-time collaboration capabilities an integral part of the Search, Profiles or K-map capabilities of Discovery Server.

With Sametime available, your users can see the online status of a user listed in the Discovery Server search results, as shown in Figure 8-8.
This is also available in the profile for users in Discovery Server as shown in Figure 8-9.

![Figure 8-9  Person profile with online status](image)

In addition to having online awareness, your users can initiate an immediate conversation with an online users as shown in Figure 8-10.
In many cases, your users will search, hoping for an immediate answer. In those cases where the search results don’t provide an answer, your users can use the real-time collaboration capabilities of Sametime to contact any user, identified in the search results, who is online.

Configuring Discovery Server to use Sametime

Enabling Sametime integration for Discovery Server is quite simple. If you do not already have a Sametime server on your network, set up a Sametime server on a machine that is separate from your Discovery Server servers.

Each person who requires Sametime access in the K-map needs a directory entry specifying the name of the Sametime server they should be connected to when opening the K-map. This entry is used by the standard K-map user interface supplied with Discovery Server, or if you use Sametime Links (see “Using the Sametime API” on page 254) if you are using a Domino directory as your directory of users for Discovery Server, the Person document in the Discovery Server Domino directory has a field on the Administration tab to specify the Sametime server (see Figure 8-11).
If you are not using the Domino directory as your primary directory, you will be using another LDAP-accessible directory. The person entity requires an attribute called SametimeServer. This attribute needs to be returned in any person entity query. This attribute needs to be set to the Domino hierarchical name of the Sametime server, for example ITSOSametime/ITSO. When a user authenticates to Discovery Server, this attribute will be used to make a connection to the Sametime server for that user.

If you are using a single Sametime server for all Discovery Server users, there is a quick and easy way of specifying the Sametime server for the users of your Discovery Server installation. In the NOTES.INI file in Lotus\DS directory of each K-map server, add the following entry:

SametimeServer=YourSametimeServer/YourOrg

For example:

SametimeServer=ITSOSametime/ITSO
If Discovery Server does not find a SametimeServer attribute in the person entity, then it will fall back to this setting. This setting is applicable whether you are using a Domino directory or another LDAP directory for user authentication.

**Using the Sametime API**

The integration of Sametime with the standard out-of-the-box K-map user interface has some limitations. For example, it is not possible for users who are only logged in to the K-map user interface to start a full Sametime instant meeting, with shared whiteboards and other collaborative tools. In order to do an instant meeting, users also need either the Sametime client installed or they must open the Java Connect client.

If you are building your own user interface to Discovery Server, you can overcome these limitations by using the Sametime Links (STLinks) toolkit. As of this writing, the toolkit is currently in beta and can be downloaded from the Lotus Developer Domain website (“Sametime Links 2.6 Toolkit”). This toolkit will be fully available and supported with the upcoming Sametime 3.0 release.

Sametime Links is a light toolkit that allows Web developers to Sametime-enable their Web pages and applications with “live names.” A simple HTML/JavaScript API allows Web developers to turn existing names into Sametime links by simply adding a few lines of HTML, without affecting the layout of the page. While rich in functionality, it is light in size – using an embedded (and hidden) applet of only 20 K to maintain the connection to the Sametime server.

For details on utilizing the Sametime links toolkit, refer to the IBM Redbook *Working with the Sametime Client Toolkits*, SG24-6666.

For more information, see:

- www.lotus.com/sametime
- www.lotus.com/sametime/developers

### 8.5.4 IBM Lotus Quickplace

IBM Lotus Quickplace is the self-service Web tool for team collaboration. Use QuickPlace to instantly create a secure and central workspace on the Web. Structured for immediate participation, teams use QuickPlace to:

- **Coordinate:** people, tasks, plans, and resources
- **Collaborate:** share ideas and discussion, resolve issues, co-author documents, exchange files, manage due diligence
- **Communicate:** actions and decisions, key findings and lessons, publish knowledge captured to a broader base of readership
How Quickplace complements Discovery Server

The self-service nature of Quickplace mean that your business community will use this as a quick and effective way to allow business teams to form, share, and collaborate on topics of mutual interest and benefit.

In doing so they will create material that has organization value. So, it is important that this information is available for all search and knowledge management initiatives.

To this end, Discovery Server 2.0 includes a specific data spider for Quickplace. This allows Discovery Server to collect and categorize information available in any Quickplace, while still preserving the security of that Quickplace.

For more information, see:

www.lotus.com/quickplace

8.5.5 IBM Lotus Domino.Doc

Domino.Doc offers complete Document Life Cycle Management – from authoring through review, approval, distribution, and archiving – for every user and every document in the organization, from narrowly-focused niche applications for small groups of specialists, to broader, scalable applications deployed enterprise-wide to every user across the organization.

How Domino.Doc complements Discovery Server

Good document management is at the heart of any organization’s successful collection, sharing, and securing of business documents. As with information stored and managed in Quickplace, information in Domino.Doc needs to be available for all search and knowledge management initiatives.

Discovery Server includes a specific spider for Domino.Doc. This allows Discovery Server to collect and categorize documents and metadata available in Domino.Doc libraries and file cabinets.

8.5.6 IBM Discovery Server Everyplace

Discovery Server Everyplace is a Palm OS application, providing a Palm conduit that is written to interface with Discovery Server via the Discovery Server API. It is an excellent example of the application of the Discovery Server API toolkit to create a truly different interface to Discovery Server.

This application allows users of Palm OS devices to pull down subsets of the K-map to their Palm device, and access the K-map of Discovery Server while disconnected from the network. Mobile users requiring access to your Discovery
Server data, or needing to search for documents and people, will benefit from the anywhere capability of Discovery Server Everyplace.

Discovery Server Everyplace provides the following capabilities to Palm OS users.

- Browse and search capabilities on a replicated subset of Discovery Server data
- Queue requests for items not available on the device for future replications
- Notification of new and updated items (documents/people/categories)
- Integration with basic Palm applications such as the address book, memo pad, and beaming

Figure 8-12 shows an example of the user interface for Discovery Server Everyplace.

![Discovery Server Everyplace](image)

**Figure 8-12** Discovery Server Everyplace

**How it works**

Discovery Server Everyplace consists of two applications, one that runs on the PDA and one that runs on the desktop. Most of the setup and maintenance work
is done on the desktop. In fact, you must install the desktop software before you can use Discovery Server Everyplace: it's the desktop application that loads the PDA application onto your hand-held device. The PDA application includes the interface for all the real user features.

The desktop application consists of three programs:

1. The conduit program runs when you synchronize the PDA. It manages the flow of information between your PDA and the desktop. It is responsible for handling new requests (such as category and document subscriptions) made on the PDA since the last synchronization, and updating your PDA information accordingly. By default, the conduit connects to Discovery Server during each synchronization session and performs a complete refresh of the local Discovery Server information.

2. The Settings dialog box lets you control how LDS Everyplace runs. The settings help determine which Discovery Server to connect to, how information is loaded from Discovery Server to the desktop, and how desktop data is loaded on the PDA.

3. The refresh client lets you maintain the “freshness” of the local Discovery Server information. The client is your primary interface for connecting to the Discovery Server to obtain the latest versions of documents, K-map, and other items. The refresh client usually runs during HotSync sessions, but you can also run it standalone by clicking the Start menu and then choosing Programs -> LDS Everyplace -> Refresh LDS Everyplace data.

Discovery Server Everyplace is available as a download at Lotus Developer Domain (www.lotus.com/ldd) in the Sandbox. Search the Sandbox for “discovery server for palm.”

An informative whitepaper on Lotus Discovery Server Everyplace can be found at:

http://www-10.lotus.com/ldd/today.nsf/lookup/DSEveryplace

8.5.7 eClassifier

The eClassifier toolkit from IBM Research can be used in an Discovery Server implementation as an alternative tool to edit your taxonomy. Compared to the standard K-map Editor it has some powerful features that enable an advanced user to work with and understand their taxonomy in additional ways. For example there are visualization options that allow you to see your documents or categories as colored dots on a scatter plot against all other documents or categories. You can even work with your K-map while in the visualization options UI by dragging and dropping items, as shown in Figure 8-13.
Using the eClassifier

eClassifier is written in Java and can run on any client that supports Java 1.3. After you've created and edited your organization's K-map you can import it into eClassifier and try some of the features discussed in this section. Be aware that the initial import will take quite a while, depending on the size of your K-map, so be patient during the initial load.

One of the biggest benefits of the tool is that you work locally, and then later save your changes to the on-line K-map if you desire. In contrast, the K-map Editor is a "live" tool that requires you to be connected to your network. Off-line editing of your K-map offers a lot of flexibility in terms of trying out different changes in your overall K-map hierarchy. If you're a consultant, this gives you the ability to show a client potential changes in their K-map without impacting their live version.
Some other interesting features of the eClassifier are:

- The Dictionary Tool allows advanced analysis of the words in your K-map and also allows you to do editing such as adding to your Stopwords list.
- One-click category creation after selecting documents is a quick way to restructure your taxonomy.
- Cohesion and Distinctness of categories are interesting measures that help you further understand your K-map.
  - **Cohesion** is the measure of how similar your categories are.
  - **Distinctness** is the measure of how different your categories are.

The scale of these two measures is 0% to 100%. An example of these measurements is in Figure 8-14.

![Figure 8-14  eClassifier Cohesion and Distinctness metrics](image-url)
eClassifier also allows you to examine and explore your categories in great detail. There are many high-level statistics that can be run to enhance the information you get from the standard Discovery Server metrics.

The Merge Class button (shown in Figure 8-14) allows you to highlight categories and merge them without the selecting, dragging and dropping, and finally removing of the category that is required using the K-map Editor.

Keyword Search is a great feature that allows you to search for all occurrences of a given keyword in a category. This allows you to create a new category for all the documents you find.

As of this writing, the eClassifier toolkit is an unsupported tool by IBM Software Group, so it should be used with discretion. Test/development environments should be used until you are accustomed to the results and effects of this tool on your K-map.

8.5.8 Atomica

Atomica Corporation (www.atomica.com) offers a set of information delivery products: Atomica Enterprise, Atomica Pro, and Atomica Personal.

These products provide servers and clients to allow backend corporate information to be indexed and made available via a simple search interface.

The client products also provide access to a range of public reference sources, such as dictionaries and thesauruses, delivered from Atomica’s public server, which are searched at the same time as corporate data.

How Atomica complements Discovery Server

Atomica has created a server add-on for Discovery Server, Atomica for Discovery Server, which allows searching of Discovery Server data at the same time as data in Atomica Enterprise and Atomica public server content sources. Available with Atomica for Discovery Server is a client that adds Discovery Server as a data source for Atomica searches, and displays search results from all Atomica sources, including Discovery Server.

Figure 8-15 shows a query entered into the Atomica Answer bar.

Figure 8-15

The result set displayed from this query shows the normal Atomica result set from the Atomica public server as well as any Atomica Enterprise server, and the
result set from Discovery Server. Figure 8-16, Figure 8-17 and Figure 8-18 show the document, people, and category results.

<table>
<thead>
<tr>
<th>Score Title</th>
<th>Date Modified</th>
<th>Author</th>
<th>File Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Re-painting Appliances</td>
<td>2000-10-03-22 27 24:000</td>
<td>Jim Schultz</td>
<td>Lotus Notes</td>
</tr>
<tr>
<td>Kitchen remodel, finally this year</td>
<td>2000-10-03-22 29 37:000</td>
<td>Suzanne Doyle</td>
<td>Lotus Notes</td>
</tr>
<tr>
<td>Actually going to happen - let's hear starting in August</td>
<td>2000-10-03-22 29 40:000</td>
<td>Suzanne Doyle</td>
<td>Lotus Notes</td>
</tr>
<tr>
<td>Heating system questions</td>
<td>2000-10-03-22 27 26:000</td>
<td>Wendy Wolff</td>
<td>Lotus Notes</td>
</tr>
<tr>
<td>Glass top ranges?</td>
<td>2000-10-03-22 29 46:000</td>
<td>Barbara Borin</td>
<td>Lotus Notes</td>
</tr>
<tr>
<td>Do refrigerators make heat?</td>
<td>2000-10-03-22 27 39:000</td>
<td>Joseph Klimowski</td>
<td>Lotus Notes</td>
</tr>
<tr>
<td>Are there any design centers around?</td>
<td>2000-10-03-22 27 22:000</td>
<td>Sean Lilly</td>
<td>Lotus Notes</td>
</tr>
<tr>
<td>Washing Machine - NEAR DISASTER</td>
<td>2000-10-03-22 27 24:000</td>
<td>Jim Schultz</td>
<td>Lotus Notes</td>
</tr>
<tr>
<td>Anyone know of any stores</td>
<td>2000-10-03-22 28 53:000</td>
<td>Mary Beth Mudduff</td>
<td>Lotus Notes</td>
</tr>
<tr>
<td>for purchasing tracked appliances with 110V only (and 220V?)</td>
<td>2000-10-03-22 29 38:000</td>
<td>Carol Easthouse</td>
<td>Lotus Notes</td>
</tr>
</tbody>
</table>

Figure 8-16  Document results
### Figure 8-17  People results

<table>
<thead>
<tr>
<th>Score</th>
<th>Name</th>
<th>Job Title</th>
<th>Location</th>
<th>Department</th>
</tr>
</thead>
<tbody>
<tr>
<td>82</td>
<td>Lynda Urgola</td>
<td>Support Manager</td>
<td>Woodford</td>
<td>Indoor/Outdoor</td>
</tr>
<tr>
<td>51</td>
<td>Carol Easthore</td>
<td>Director of Support</td>
<td>Cambridge</td>
<td>Indoor/Outdoor</td>
</tr>
<tr>
<td>76</td>
<td>Rosa Marz</td>
<td>Phone Support Tech</td>
<td>Cambridge</td>
<td>Indoor/Outdoor</td>
</tr>
<tr>
<td>76</td>
<td>Susan Devie</td>
<td>Phone Support Tech</td>
<td>Woodford</td>
<td>Indoor/Outdoor</td>
</tr>
</tbody>
</table>

Lotus Discovery Server

People whose profile contains "appliances"

Click a person to bring up their profile and see more information.
In addition to providing the Answer Bar to allow queries using Atomica, Atomica also integrates with any text display or editing tool, such as word processors, and with browsers.

This allows you to Alt-Click on any word that is displayed, and Atomica will automatically initiate a search, and display the results set. This is useful if you are reading a document and the meaning or context of a word is not clear. Atomica allows you to find a definition of the word, and people who know about the word or related topics.

Figure 8-19 provides an example, where the word “plug” in the sentence in a word processor is used as the search criteria, and Alt-Clicking on the word plug produces the displayed Atomica results set, including results from Discovery Server.
Installation

The Atomica server add-on is installed on the Discovery server. An overview of the files that are installed and modified follows.

The following files are installed during the installation process:

- LDSapi.jar, HTTPClient.jar, jakarta-regexp-1.2.jar, and atomic-Discovery Server.jar are installed into distaff
- atomica.properties is installed into ds\data\atomica
- setup.exe and setup_info.ini are installed into ds\data\atomica

These lines added to the file ds\data\servlets.properties:

```java
servlet.query.code=com.atomica.servlets.query
servlet.ATlookup.code=com.atomica.servlets.query
servlet.ATpassthrough.code=com.atomica.servlets.query
```
These entries added to line JavaUserClasses in \ds
notes.ini:

JavaUserClasses=... existing classes
...;java\LDSapi.jar;java\HTTPClient.jar;java\jakarta-regexp-1.2.jar;java\atomica-Discovery Server.jar

In order to enable use of the Discovery Server system through Atomica, users need to install the Atomica Answer Window client software that is provided with Atomica for Discovery Server. If a user has an existing copy of the Atomica Answer Window, it must first be uninstalled.

The client setup program can be found in the Atomica directory (ds\data\atomica), along with a configuration file, setup_info.ini. Clients can be installed from this directory directly, or you can move both files to a more convenient shared directory. Both files must be in the same directory in order for the installed clients to point to the correct server.

Alternatively, you can E-mail the client to your users, and provide them with the name of the server that they should point the client to during installation. The server should be fully qualified (server.domain.com).

The Atomica Answer Window client uses the Atomica servlet installed on the Discovery server to query Discovery Server data and Atomica public server data in a single query.

If an Atomica Enterprise server is also installed, you can modify the atomica.properties file in Lotus\DS\Data\Atomica to include a reference to the Atomica Enterprise server. The atomica servlet on Discovery Server will then make a request to the enterprise server to return data from the enterprise server and the public server. The Atomica servlet will then consolidate results with results from Discovery Server and present them via the Atomica Answer window.

For more information, see:

www.atomica.com

8.5.9 Kamoon

Kamoon Inc provides Kamoon Connect, an enterprise expertise management (EEM) system. Kamoon’s EEM technology matches inquiries to the right expert, facilitates the right interaction process through its action-oriented workflow,
captures questions and answers in a FAQ database, and provides measurement tools for enhancing the entire question and answers process.

A Web-based interface gives a user the ability to enter a query as a complete sentence. Kamoon parses this question to get an understanding of the nature of the question, including doing an analysis of the semantics (meaning of the words) of the sentence.

Kamoon will then search an existing database of PAQs (Previously Asked Questions), created and maintained in Kamoon, for an answer. If an answer is not found, Kamoon searches for a list of experts who could answer the question and begins a workflow to get an answer from one of these experts.

Once the question is answered, the requester is able to classify the value of the answer and the question and answer are added to the PAQ database.

**How Kamoon complements Discovery Server**
Kamoon uses Discovery Server for dynamic profiling. The integration works by Kamoon sending a text string command to the Lotus Discovery Server using the Discovery Server API. Discovery Server matches the text with the stored expert profiles and submits a list of matching experts back to the Kamoon system. The Kamoon EEM policy engine processes the expertise information from Discovery Server, together with the company's business policies, and initiates the appropriate workflow.

At the end of the Q and A process that Kamoon manages, it returns the value information about the answer given to the metrics engine of Discovery Server by calling the Metrics servlet.

For more information, see:

www.kamoon.com
Maintaining a Discovery Server environment

By this point you have probably managed to get a Discovery Server installation up and running, and have gotten your first set of users using the facilities that Discovery Server provides, in a way appropriate to their needs.

So now the question is, what do you do if something goes wrong, or more to the point, what do you do before something goes wrong?

In this chapter we discuss system troubleshooting, system backups, and system reconfiguration/modifications. You should use the generic material presented here to build policies and procedures appropriate to your own particular environment.

To gain the greatest benefits from this chapter, you need to be familiar with the Discovery Server installation in your organization, how it was deployed, and how it is being used. A thorough reading of the rest of this book prior to reading this chapter should assist in this.
9.1 Server recovery and troubleshooting

When problems arise, the pressure to get the server back up and running can be a source of conflict for the Discovery Server administrator who, on the one hand is under pressure to get everything running, and on the other, wants to capture important diagnostic information to allow full resolution of the problem. To strike a balance between these competing requirements, administrators need to have a good understanding of how the various components of the Lotus Discovery Server work together. We recommend that administrators take the time to read Chapter 2, “The Discovery Server architecture and key components” on page 13, which explains the server architecture in detail.

We also encourage all personnel involved in the administration and management of the Lotus Discovery Server to become familiar with the basic concepts and architecture of a Lotus Discovery Server environment. With such a foundation in place, administrators, managers, taxonomy editors, IT support staff, and others will be able to communicate; and with a common understanding of how the Lotus Discovery Server has been deployed, they will be able to more effectively contribute to diagnosing and troubleshooting server problems.

9.1.1 Always read the release notes first

We strongly recommend that all administrators read the online release notes (readme.chm) that come with the Lotus Discovery Server product, or are available online at the Lotus Developers Domain as described in “Product documentation” on page 297.

The readme file is a compiled HTML help file that can be launched by double-clicking on it in Windows Explorer. It contains critical release notes you need to consider as part of your Lotus Discovery Server deployment.

Inexperienced system administrators will probably find the Lotus Discovery Server is a technically demanding product to install and configure, and they are strongly advised to allocate plenty of time to read all the online documentation and generally familiarize themselves with the way the Lotus Discovery Server works. Experienced administrators are also advised to resist the temptation to launch straight into the installation without reading the online documentation.

9.1.2 Why you need a Server Recovery Plan

Dealing with server problems is best managed if the Lotus Discovery Server administrator can adopt a calculated, methodical approach based on sound problem management practices. To this end, one of the key themes emphasized throughout this chapter is the value of being able to refer to a Server Recovery
Plan (SRP) that provides operational guidelines for dealing with IT infrastructure problems.

Well-prepared administrators usually have a set of procedures or a checklist in place to promote efficient resolution of problems during the server recovery process. These procedures should be consistent with the guidelines laid down in the Server Recovery Plan. In this way, administrators can effectively initiate appropriate responses and actions in the event of a server failure.

Having access to a Server Recovery Plan can also be beneficial in situations where:
- An experienced administrator is unavailable or leaves the company
- Server recovery is being handled by an inexperienced administrator
- New administrators are being trained on server recovery procedures

The following list represents some broad areas of server operation your system administrators would be expected to deal with in the event of a server outage or problem. Your Server Recovery Plan should provide a framework which outlines what actions and responses are appropriate for managing problems in your IT infrastructure.
- Dealing with conflicts of interest
- Meeting Service Level Agreements (SLAs) specified for various components in your IT environment
- Managing server outages
- Recording/reporting server outages
- Monitoring and maintaining server performance
- Problem escalation

As part of your Lotus Discovery Server deployment, remember to update your response plan to accommodate specific requirements for monitoring, maintaining, and troubleshooting your server infrastructure.

### 9.1.3 Dealing with conflicts of interest

When a server problem is reported, the Lotus Discovery Server administrator has to initially make a quick assessment of:
- What is the impact on users?
- How serious is the problem?
- What service level agreements apply in this situation?
Figuring out the impact on users can be a tricky problem for the Lotus Discovery Server administrator. On a Lotus Discovery Server, there are no readily visible statistics the administrator can view on the server console. Domino administrators who are familiar with issuing a `show users` command at the Domino server console will find the same command on a Lotus Discovery Server console will not list active users because HTTP connections do not create persistent sessions. Instead, HTTP logs in either the domlog.nsf or standard HTTP log files should be consulted to find out normal users and usage trends.

Metrics reports can be generated to provide some post-analysis information on who is using the system. These reports can show you who is using the K-map and Search. By looking at these reports, the administrator can see who has been active on the system.

In any failure, the first thing to check out is whether the K-map is still available for user access. Loss of access to the K-map is a serious problem because users will not be able to:

- Submit searches
- View or navigate around the K-map
- Locate expertise
- View affinities
- Edit the K-map

If the K-map is unavailable, it probably means there is a serious problem on the primary server. This, in turn, affects all secondary servers because most processes running on secondary servers rely on communication with the primary server.

Checking on the status of the K-map is easy; just try to submit a search or use the K-map Editor to open the K-map.

In terms of the impact on users, loss of the primary server is the most serious failure an administrator will have to deal with. If SLAs state your Lotus Discovery Servers should be recovered after 30 minutes (anything less would be difficult to achieve), the administrator will be under severe pressure to collect all important crash information and get the server back up within this time.

Loss of a secondary server is unlikely to have a direct impact on users, and is more of an operational issue for the Lotus Discovery Server administrator. In this case, a different set of SLAs should apply. Knowledge of how various Discovery Server services are distributed across secondary servers is essential. For example, K-map Indexing services user searches, whereas spidering a data repository is a background task that has no direct impact on user activity. Thus, if
a secondary server is running user oriented services such as indexing, its
downtime will impact users.

Service Level Agreements (SLAs) are a constant reminder to administrators that
the system exists for the benefit of users. In negotiating a reasonable set of
Service Level Agreements for your Lotus Discovery Servers, consider the
following points:

- A Discovery Server outage is not a total loss of service for users.
- Not all users in your organization will access the Lotus Discovery Server
  environment, and of those that do, an even smaller percentage of the user
  base are active at any point in time.
- Loss of a secondary server may have no direct impact on users.
- Loss of a primary server will probably have an impact on users.

On occasion, administrators may need to override SLA requirements to allow
capture of critical system information for problem resolution purposes. In this
case, short-term commitment to get the system up and running for user access is
waived in favor of attaining long-term system stability through detailed
investigation of the problem. In serious cases, the administrator may be under
instruction from the vendor to collect important system crash information to help
diagnose the problem.

### 9.1.4 Types of crashes

For organizations that have a Server Recovery Plan in place, guidelines for
managing server problems will help administrators draw up appropriate
procedures or checklists for recovering a server. For example, administrators will
be expected to recover a server within pre-negotiated SLAs, log server outages
and escalate serious problems to designated authorities/managers. The amount
of pressure an administrator is under will depend to some extent on how well
prepared they are for dealing with a each type of server crash.

In the following sections we describe three types of server failures that the
administrator is likely to encounter. Included under each failure type are
symptoms you might see, what problem(s) the user sees, and suggested
responses the administrator might adopt. For all three failure types, the
administrator must strike a balance between loss of service for users and time
required to diagnose, collect crash information, and recover the system.

**Degraded Service**

In this scenario, the user still has access to all parts of the system, but
experiences poor response times when accessing the server. For example, the
K-map takes a long time to open.
Server symptoms
A degraded service is typically a result of poor system performance when server resources are unable to service demand. This is usually caused by insufficient memory, memory leaks, disk I/O bottlenecks, excessive processor demand, over-committed network utilization, or incorrect configuration of the system.

Administrator response
The administrator has the opportunity to collect critical system information while the system is still running. Pressure to shut down and restart the server depends on how the slow performance is affecting users. Important evidence can be captured while users still have access to the system.

Examine system performance using tools like Windows Performance Monitor, Windows Task Manager, or 3rd party performance tools. If necessary, upgrade hardware based on analysis of performance monitoring data.

Partial failure
A user reports a problem with accessing certain parts of the system, for example, Sametime awareness is not working properly.

Server symptom
One or more tasks have generated an exception or abnormal termination event which renders the tasks incapable of servicing or processing data requests. Severity of this crash will depend on which tasks have stopped and how dependent users are on the tasks.

Administrator response
Opportunities for collecting critical system information are reduced because users may not be able to tolerate loss of functionality. There is greater pressure to stop and restart the Lotus Discovery Servers. Much depends on what part of the system has failed.

Any system failure on the primary server is probably intolerable from both a user and system administration perspective. Failure on a secondary server such as spidering a data repository or updating the K-map Index could be tolerated pending an overnight reboot of the server. Where users are not directly accessing a secondary server, an immediate reboot of the server is possible.

Complete failure
A user reports that the system is unavailable or not responding.
**Server symptom**
Lotus Discovery Server is no longer responding. The Task Manager shows one or more tasks as no longer responding. The server console possibly is hung; the Discovery Control Center is not responding.

**Administrator response**
In this circumstance, an administrator is probably under extreme pressure from users to get the system back up and running as soon as possible. Any opportunities for collecting critical system information are under severe time constraints. The administrator needs to efficiently capture the minimum amount of crash information required in the Server Recovery and Troubleshooting response plan.

### 9.1.5 Using Windows tools to diagnose a problem

Windows NT and Windows 2000 both have Task Manager and Performance monitoring tools installed as part of the original operating system. These tools, in the hands of a skilled administrator, can be used to effectively monitor and diagnose most problems that occur on a Lotus Discovery Server.

Administrators should at least be familiar with using Task Manager to examine resource usage and processes running on the server. Advanced administrators will be able to use the Performance monitoring tool to display graphs of system performance and generate logs for analyzing system performance.

We recommend that administrators compile system performance benchmarks for each Lotus Discovery Server to establish a baseline or reference point of normal server operation. Compiling such a set of benchmarks puts you in a strong position to analyze, monitor, and even predict how changes on your system impact server performance. For example, predicting the impact of spidering a new data repository is possible if performance data has been previously captured for a similar data repository.

**Windows Task Manager**
Windows Task Manager can be used to capture real-time resource usage on your Lotus Discovery Server. After loading Task Manager and selecting the Processes tab, we recommend you add the following performance counters to the default set of columns displayed:

- Virtual Memory Size
- I/O Reads
- I/O Writes
- I/O Read Bytes
- I/O Write Bytes
Figure 9-1 shows how the Task Manager can be used to display useful information on various processes running on your Lotus Discovery Server. This can be achieved with very little effort by the administrator.

<table>
<thead>
<tr>
<th>Image Name</th>
<th>CPU Time</th>
<th>Mem Usage</th>
<th>VM Size</th>
<th>I/O Reads</th>
<th>I/O Writes</th>
<th>I/O Read Bytes</th>
<th>I/O Write Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>db2syscs.exe</td>
<td>00.00:05</td>
<td>179,554K</td>
<td>355,196K</td>
<td>1,345</td>
<td>676</td>
<td>24,716,999</td>
<td>375,909</td>
</tr>
<tr>
<td>rhtdp.exe</td>
<td>00.00:11</td>
<td>4,840K</td>
<td>30,144K</td>
<td>15,063</td>
<td>685</td>
<td>27,147,055</td>
<td>1,580,664</td>
</tr>
<tr>
<td>ncmserv.exe</td>
<td>00.00:07</td>
<td>3,284K</td>
<td>30,120K</td>
<td>419,360</td>
<td>12,094</td>
<td>16,723,493</td>
<td>254,041</td>
</tr>
<tr>
<td>natg.exe</td>
<td>00.00:00</td>
<td>3,724K</td>
<td>21,228K</td>
<td>75,413</td>
<td>2,621</td>
<td>268,005,008</td>
<td>49,508</td>
</tr>
<tr>
<td>rlmimap.dll.exe</td>
<td>00.00:02</td>
<td>2,038K</td>
<td>14,824K</td>
<td>29,882</td>
<td>8,468</td>
<td>28,945,225</td>
<td>4,984,850</td>
</tr>
<tr>
<td>rmetrics.exe</td>
<td>00.00:01</td>
<td>2,340K</td>
<td>12,984K</td>
<td>3,806</td>
<td>95</td>
<td>3,665,995</td>
<td>92,744</td>
</tr>
<tr>
<td>rupdate.exe</td>
<td>00.00:00</td>
<td>2,056K</td>
<td>12,296K</td>
<td>1,029</td>
<td>135</td>
<td>2,095,052</td>
<td>188,704</td>
</tr>
<tr>
<td>rlmexplorer.dll.exe</td>
<td>00.00:00</td>
<td>1,684K</td>
<td>11,888K</td>
<td>224</td>
<td>13</td>
<td>224,780</td>
<td>868</td>
</tr>
<tr>
<td>ndsched.exe</td>
<td>00.00:02</td>
<td>2,516K</td>
<td>10,644K</td>
<td>13,539</td>
<td>1,435</td>
<td>15,950,536</td>
<td>13,030,904</td>
</tr>
<tr>
<td>psp.exe</td>
<td>00.00:02</td>
<td>1,592K</td>
<td>4,472K</td>
<td>301</td>
<td>104</td>
<td>516,723</td>
<td>22,361</td>
</tr>
<tr>
<td>db2syscs.exe</td>
<td>00.00:00</td>
<td>6,964K</td>
<td>7,055K</td>
<td>23</td>
<td>16</td>
<td>205,995</td>
<td>834</td>
</tr>
<tr>
<td>db2syscs.exe</td>
<td>00.00:00</td>
<td>7,212K</td>
<td>7,052K</td>
<td>57</td>
<td>24</td>
<td>396,716</td>
<td>26,020</td>
</tr>
<tr>
<td>resv.exe</td>
<td>00.00:08</td>
<td>5,552K</td>
<td>5,972K</td>
<td>4,126</td>
<td>535</td>
<td>8,901,323</td>
<td>2,164,172</td>
</tr>
<tr>
<td>WINLOGON.exe</td>
<td>00.00:00</td>
<td>3,104K</td>
<td>5,889K</td>
<td>494</td>
<td>123</td>
<td>388,273</td>
<td>7,287</td>
</tr>
<tr>
<td>ldap.exe</td>
<td>00.00:00</td>
<td>1,812K</td>
<td>5,644K</td>
<td>1,675</td>
<td>8</td>
<td>3,421,557</td>
<td>429</td>
</tr>
<tr>
<td>ndisp.exe</td>
<td>00.00:00</td>
<td>1,504K</td>
<td>5,288K</td>
<td>57</td>
<td>46</td>
<td>56,775</td>
<td>85,160</td>
</tr>
<tr>
<td>ntfs expérience</td>
<td>00.00:02</td>
<td>6,920K</td>
<td>5,023K</td>
<td>28,053</td>
<td>97</td>
<td>1,510,656</td>
<td>1,407</td>
</tr>
<tr>
<td>explorer.exe</td>
<td>00.00:05</td>
<td>3,040K</td>
<td>4,940K</td>
<td>706</td>
<td>147</td>
<td>1,223,000</td>
<td>13,409</td>
</tr>
<tr>
<td>rmsg.exe</td>
<td>00.00:01</td>
<td>1,004K</td>
<td>3,549K</td>
<td>1,023</td>
<td>171</td>
<td>4,937,214</td>
<td>61,596</td>
</tr>
<tr>
<td>rrouter.exe</td>
<td>00.00:00</td>
<td>1,100K</td>
<td>3,123K</td>
<td>101</td>
<td>16</td>
<td>145,960</td>
<td>1,702</td>
</tr>
<tr>
<td>rnetsvc.exe</td>
<td>00.00:00</td>
<td>3,956K</td>
<td>2,924K</td>
<td>120</td>
<td>23</td>
<td>106,059</td>
<td>27,996</td>
</tr>
<tr>
<td>services.exe</td>
<td>00.00:01</td>
<td>5,012K</td>
<td>2,689K</td>
<td>15,078</td>
<td>15,340</td>
<td>2,041,253</td>
<td>1,669,140</td>
</tr>
<tr>
<td>db2id.exe</td>
<td>00.00:00</td>
<td>3,432K</td>
<td>2,940K</td>
<td>21</td>
<td>5</td>
<td>8,424</td>
<td>268</td>
</tr>
<tr>
<td>rmng.exe</td>
<td>00.00:00</td>
<td>1,932K</td>
<td>2,380K</td>
<td>1,055</td>
<td>45</td>
<td>2,269,432</td>
<td>88,147</td>
</tr>
</tbody>
</table>

In looking at the processes running on this example server, 3 points of interest can be found. These are:

1. The db2syscs.exe process is using a relatively large amount of virtual memory.
2. The ncmserv.exe process does a lot of I/O Reads which fetch small amounts of data per read.
3. The natg.exe process reads significantly more bytes than other tasks and performs fewer I/O Reads than the ncmserv.exe process.

Switching to the Performance tab within Task Manager can provide useful information on CPU and memory usage.
In this example, you can immediately see total memory usage exceeds the total physical RAM installed on the server. Your initial reaction might be to add more physical RAM to avoid excessive memory paging. However, beware of jumping to a quick conclusion. On closer examination of the graph, you will see in the Physical Memory box there is still 182044 bytes free. If you refer back to the previous figure, which lists processes running on the server, you will find the DB2syscs.exe process is consuming both physical and virtual memory.

The point to keep in mind here is that basic tools like Task Manager and the Performance monitor can provide, at little cost, a lot of valuable information on how your system is performing.
Windows Performance monitor

To examine server processes on a timeline basis, the Windows Performance monitoring tool can be used to produce graphical and log output. Logged output can be saved in a variety of file formats and analyzed later using spreadsheet tools and statistical analysis packages.

Discovery Server processes

If you are monitoring the Discovery server processes via Windows Task Manager or Performance monitor, the Discovery Server-related tasks identified in Table 9-1 should be carefully monitored.

Table 9-1  Discovery Server processes to monitor

<table>
<thead>
<tr>
<th>Name in DS console</th>
<th>Formal name</th>
<th>Name in Task Manager</th>
</tr>
</thead>
<tbody>
<tr>
<td>WebSpider</td>
<td>Web spider</td>
<td>nwebspider.exe</td>
</tr>
<tr>
<td>FileSystemSpider</td>
<td>File system spider</td>
<td>nfilesyspider.exe</td>
</tr>
<tr>
<td>NotesSpider</td>
<td>Notes spider</td>
<td>nnotesspider.exe</td>
</tr>
<tr>
<td>KmapIndexing</td>
<td>K-map Indexing</td>
<td>nkmapindex.exe</td>
</tr>
<tr>
<td>ATG</td>
<td>K-map building</td>
<td>natg.exe</td>
</tr>
<tr>
<td>metrics</td>
<td>Metrics</td>
<td>nmetrics.exe</td>
</tr>
<tr>
<td>Scheduler</td>
<td>Scheduler</td>
<td>ndssched.exe</td>
</tr>
<tr>
<td>DS (or Content Map)</td>
<td>Discovery data store</td>
<td>ncmserv.exe</td>
</tr>
</tbody>
</table>

9.1.6 Capturing crash information

It is the job of the administrator to collect as much useful information as possible in the event of a server failure. The ability to collect crash information will depend on a number of factors, most of which are under the control of the Lotus Discovery Server administrator.

As mentioned in 9.1.3, “Dealing with conflicts of interest” on page 269, the administrator is under considerable pressure to get the system back up and running. New or inexperienced administrators should familiarize themselves with response guidelines specified in the Server Recovery Plan to ensure appropriate procedures have been followed to recover the server.

Remember, the server recovery process doesn’t finish when the server is back up and running, it includes recording, reporting, and post-crash analysis of information collected during the recovery process.
Don’t destroy the evidence

Rigorous recording of crash information and capture of evidence can be a time-consuming process. However, failure to record crash information can lead to repeated server crashes and prolonged instability for users because important evidence has been ignored or lost.

In some situations, the administrator may choose to deliberately delay shutting down and restarting a server to allow critical system information to be captured for deeper diagnosis of an ongoing problem. For example, evidence on how much memory a particular process is consuming may mean delaying a server shutdown to allow the Task Manager or the Performance monitor to be used to examine resource usage.

Rebooting a server without first analyzing what is currently happening on the server can prove to be a frustrating exercise for all concerned. Critical diagnostic information can be lost and you may be setting yourself up for a lengthy period of server instability.

Screen dumps, errors, and logs

The importance of accurately capturing crash information cannot be over-stated. When writing down error messages from the server console, make sure the error message is recorded exactly as it is presented on the screen (including upper/lower case characters). The ability of a support analyst to examine and successfully diagnose a problem reported by a customer relies heavily on the accuracy and completeness of the information submitted. Descriptions of the sources of information you should consider collecting follow.

Screen dumps

Screen dumps are one of the most effective ways of accurately capturing the current status of your system. On most servers you can press one or two keys to dump screen information to the clipboard.

Table 9-2  Keys to dump screen output to the clipboard

<table>
<thead>
<tr>
<th>Keys</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Print Screen</td>
<td>Captures the entire screen</td>
</tr>
<tr>
<td>Alt + Print Screen</td>
<td>Captures the currently active window</td>
</tr>
</tbody>
</table>

* Actual Print Screen Keys may vary depending on how your system is configured.

Initially, the screen output is saved to the system clipboard. From there you can paste the contents of the clipboard into Microsoft Paint and save the output as an image file.
The Lotus Discovery Server console

The contents of the Lotus Discovery Server console can be captured by marking and copying the contents of the console window (see Figure 9-3). If you run the server console in full screen mode, press the Alt and Enter keys together to switch to window mode. The copied information is saved to the clipboard as text. From there you can paste the contents of the clipboard into Notepad and save the output as a text file.

![Figure 9-3 How to mark the contents of the server console](image)

**Tip:** You can increase the amount of information stored in the Lotus Knowledge Discovery System console by increasing the height of the screen buffer size. This is illustrated in Figure 9-4.

![Figure 9-4 Increasing the screen buffer size](image)
Windows dialog boxes
Error messages displayed in dialog boxes can be easily captured to a file. First, switch focus to the dialog box by clicking on its title bar. Now dump the contents of the active window to the clipboard by pressing Alt and Print Screen together. Now save to a file using Microsoft Paint.

Task Manager
Task Manager can be loaded to view system resource usage and process information. This tool can be used to view the current status of your Lotus Discovery Server and diagnose resource usage problems (for example, running out of memory). Information displayed in the Task Manager window can quickly be dumped to the clipboard by pressing the Alt and Print Screen keys together.

Log output
The Lotus Discovery Server writes log output to a file called discoverylog.nsf. To accurately capture server activity, consider making a copy of this file to send to your support analyst. Be careful, though; this file can be very large.

Tip: To manage the size of the discoverylog.nsf file, write a simple agent which selectively purges documents based on your criteria (for example, all entries older than 90 days).

The Discovery Control Center provides numerous links into the Discovery log since it's not designed to be viewed directly (although this is not prevented). You'll find "View <service name> log for this repository" links in the Status sections of every repository form that has been spidered.

The Discovery services also post logging information to the local Notes Log (log.nsf) on every Discovery Server where Domino log messages get posted. Only Start, Warning (not associated with a specific repository), Fatal Error (not associated with a specific repository), and Stop messages get posted to the Notes log, and the message text that gets posted to this log file is mirrored in the server console window.

Server configuration settings
The Lotus Discovery server document contains configuration settings which control access to the server, security settings, servlets, http access, and so forth. Information in this document can easily be cut and pasted into a file to send to a support analyst.
**discoveryadmin.nsf**
The discoveryadmin.nsf database stores a lot of useful information about how you have configured your Lotus Discovery Server system. A support analyst can examine this file to check configuration settings and see what data repositories you are spidering.

**XML queues**
Your support analyst may ask you to examine various XML queues to determine the status of your system. Using Windows Explorer, you can list which work queues reside on your servers. Displaying the file size and last modified date/time can reveal important information about the current status of you system (Figure 9-5 on page 283 as an example of this). If necessary, you can screen dump the contents of Windows Explorer to the clipboard and save the output to a file.

**Note:** All XML work queues are initialized as 4 KB files.

**Recording incidents**
Guidelines for determining what information should be recorded in the event of a system failure should be drawn from your Server Recovery Plan. Following is an example of the type of information we suggest you record in a server incident report. Ideally this information should be entered into a database (such as Notes or Quickplace) for easier searching, viewing, and analysis of server problems.

<table>
<thead>
<tr>
<th>Reported by:</th>
<th>Andy Crashman</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server Name:</td>
<td>LDS01/Atlanta</td>
</tr>
<tr>
<td>Date of incident:</td>
<td>04/13/2001</td>
</tr>
<tr>
<td>Time incident reported:</td>
<td>10:15 AM</td>
</tr>
<tr>
<td>Time incident resolved:</td>
<td>10:35 AM</td>
</tr>
<tr>
<td>Status:</td>
<td>Resolved</td>
</tr>
<tr>
<td>Crash symptoms:</td>
<td>Discovery Control Center reports Notes Spider no longer responding. Started spidering at 8:30am. Lotus Discovery Server still running and able to respond to console commands. XML queues on Primary Discovery Server not receiving any new data.</td>
</tr>
</tbody>
</table>
| Files/Servers involved: | Notes database - sales2000.nsf  
Server - Marketing/SRV/Atlanta.  
Spider - “Sales2000” on LDS01/Atlanta |
We recommend establishing some form of incident reporting system which logs all problems found on your servers. Maintaining an audit log of all server incidents can provide valuable context for ongoing server problems which, in isolation, provide no clues as to why the server crashed. Consider the following scenario which illustrates this point.

**Example scenario**

You are investigating a problem where one of your Lotus Discovery Servers crashes after nearly 200 hours of continuous operation. The problem has occurred five times over the last two months. In looking back through the incident reports you notice another incident report for the same server is always raised the day before the Lotus Discovery Server crashes. It states the weekly backup of the server failed.

Here is an example of how context has been added to the problem. A pattern starts to emerge where Lotus Discovery Server crashes are preceded by a weekly backup failure.

Further investigation reveals the Lotus Discovery Server is always rebooted after a successful completion of a weekly backup. Further testing reveals a memory leak is introduced on the Lotus Discovery Server after the weekly backup fails. 24 hours later, the Lotus Discovery Server crashes when all available memory is consumed.

In this example, the root cause of the problem could have easily remained undetected if the problem had not been viewed in context. The administrator could have (incorrectly) concluded the Lotus Discovery Server had a slow memory leak which could be “managed” by rebooting the server every Sunday.
night (this would only work if the weekly backup was run earlier in the day). By examining a complete log of incident reports for the server, an association between failed backups and Lotus Discovery Server crashes could be made.

**Location of Lotus Discovery Server system files**

In developing a server recovery response plan for your Lotus Discovery Servers, it is a good idea to compile a checklist or table of where important system information is stored. This list can then be used by the administrator to efficiently locate and capture relevant information in the event of a system failure. The table below lists the key files that are installed as part of a standard Lotus Discovery Server implementation that could be important in any server recovery efforts.

**Note:** In a distributed implementation of the Lotus Discovery Server, some of these files may be located on the secondary rather than the primary Discovery Server.

<table>
<thead>
<tr>
<th>Type</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Notes Databases</td>
<td><code>\lotus\ds\data\dashboard.nsf</code></td>
</tr>
<tr>
<td></td>
<td><code>\lotus\ds\data\discoveryadmin.nsf</code></td>
</tr>
<tr>
<td></td>
<td><code>\lotus\ds\data\discoveryserver.nsf</code></td>
</tr>
<tr>
<td></td>
<td><code>\lotus\ds\data\indexmap.nsf</code></td>
</tr>
<tr>
<td></td>
<td><code>\lotus\ds\data\people.nsf</code></td>
</tr>
<tr>
<td></td>
<td><code>\lotus\ds\data\peopleq.nsf</code></td>
</tr>
<tr>
<td>Logs</td>
<td><code>\lotus\ds\data\log.nsf</code></td>
</tr>
<tr>
<td></td>
<td><code>\lotus\ds\data\discoverylog.nsf</code></td>
</tr>
<tr>
<td></td>
<td><code>\lotus\ds\db2log.log</code></td>
</tr>
<tr>
<td></td>
<td><code>\lotus\ds\data\cmap.log</code></td>
</tr>
<tr>
<td></td>
<td><code>\lotus\ds\data\metrics.log</code></td>
</tr>
<tr>
<td></td>
<td><code>\lotus\ds\data\db\db2\db2diag.log</code></td>
</tr>
<tr>
<td>.ini</td>
<td><code>\lotus\ds\Notes.ini</code></td>
</tr>
<tr>
<td></td>
<td><code>\lotus\ds\formats.ini</code></td>
</tr>
<tr>
<td></td>
<td><code>\lotus\ds\keyview.ini</code></td>
</tr>
<tr>
<td>XML work queues (see Figure 9-5 on page 283)</td>
<td><code>\lotus\ds\data\</code></td>
</tr>
<tr>
<td>DB2 databases</td>
<td><code>\DB2\*.*</code></td>
</tr>
<tr>
<td></td>
<td><code>\DB2CTLSV\*.*</code></td>
</tr>
<tr>
<td>Discovery Server Servlets</td>
<td><code>\lotus\ds\data\domino\servlet</code></td>
</tr>
<tr>
<td></td>
<td><code>\lotus\ds\java\</code></td>
</tr>
</tbody>
</table>
9.1.7 Analyzing operation of CMAP

Discovery Server uses IBM DB2 to store information about the data it manages. Occasionally, the CMAP task will report errors, or you will see unexplained server lockups. The following procedure can be used to create event monitor logs for DB2 which provide useful information in troubleshooting any errors.
Setting up an Event Monitor for DB2
When trying to find the course of events (locking/unlocking, querying tables, and so forth) that lead to a DB2 deadlock, use the DB2 Event Monitor as follows:

1. Open DB2 Event Monitor by selecting **Start -> Programs -> IBM DB2 -> Event Monitor.**
2. When asked to specify what database, type **cmap**, then click **OK**.
3. In the Event Monitor window, choose the **Event Monitor** menu dropdown and select **Create**.
4. Name the monitor something like **"deadlock"** (This is a directory name; you can also choose where this directory resides. Choose a directory with sufficient available disc space.)
5. Choose the following event types by checking the box for Tables, Deadlocks, Transactions, and Statements.
6. Check the Start now and Restart automatically checkboxes and click **OK**. The monitor is now running.

**Important:** Turn off Restart automatically when you have the information you need to avoid consuming large amounts of disk space.

Viewing Event Monitor output
There are two ways to view the output from the event monitor: from a DB2 user interface, or by creating a text file.

**From a DB2 user interface**
You can view the event monitor output through the DB2 UI provided; however, it can be quite sluggish, especially if you have a lot of logged information. You can view the output via the UI by right-clicking the event monitor in the Event Monitor window and choosing **View Event Monitor Files.** From here, select from a list of event start times (a new one will appear for each restart of the monitor). Right-click on this and choose **Open As**, then select from the list (Connections, Deadlocks, Deadlock connections, Transactions, and so forth). From here, you can continue to right-click on individual connections, deadlocks, and so forth, and drill down for more information on each, all the way down to specific SQL statements issued to DB2.

**Attention:** Note that this monitor can quickly consume disk space (multiple files in excess of 800 Mb each) with the event monitor files that it creates, so you only want to run it while monitoring closely for your specific CMAP error, then turn off the monitor when finished. For this reason, this procedure should only be used to log a repeatable error in a known timeframe.
**Create a text file**

Because moving through large output logs is time consuming, the following is an alternative method. This method is also appropriate if you need to send files to IBM Support for analysis.

Create a text file from the individual *.EVT files that you want to see from a specific date/time using these steps:

1. Find the CMAP DB Exception messages in your Discovery Server log and note the exact time that the exception happened.
2. Open a DB2 Command Window by selecting **Start -> Programs -> IBM DB2 -> Command Window**.
3. Go to the output directory that you named for your Event Monitor. If you did not specify the directory specifically, the default location is `c:\DB2\NODE00001\DB2EVENT\eventmonitorname`.
4. Create a subdirectory in order to copy specific .evt files. (We suggest using the date/time to name the subdirectory.)
5. Type `dir *.evt` to see all the individual event logs (all about 800 Mb) and copy all the .evt files that correspond to the date/time of the exception into the new subdirectory.
6. Change to the subdirectory and rename the event files so that they are in sequence, like the following:
   - `00000000.EVT`
   - `00000001.EVT`
   - `00000002.EVT`
7. Now type the following command, which will redirect the output of all these files to a text file that you name.

   ```
   db2evmon -path c:\your_output_file_subdirectory_path > your_output_filename.txt
   ```

The resulting text file now contains all the transaction, deadlock, tables, and statement data for the specific time period.

### 9.2 Ensuring successful Discovery Server backups

Existing backup strategies, which have been developed for Domino servers, will need to be modified to cover additional software components that are installed as part of a Discovery Server installation. The main difference you need to allow for is backup of the DB2-based K-map that is installed on the primary Lotus Discovery Server.
For a distributed server implementation, the backup strategy is complicated by the fact that you have to identify what Discovery services have been installed on your secondary servers. You also have to decide whether you want to perform full system backups of your servers (the simplest option), or selective backups to minimize downtime and storage costs of your backup media.

A key issue to keep in mind when developing your backup strategy is the amount of time the primary Lotus Discovery Server will be down. This is important because many of the Discovery Server secondary servers will not happily tolerate the loss of access to the primary server. The rule of thumb to apply in any strategy is primary server down last and back up first.

9.2.1 Developing a backup strategy

To establish what type of backup strategy is most suitable for your Discovery Server infrastructure, try answering the following questions about the status of your current backup strategy and its capacity to meet your Discovery Server backup requirements. Your answers to these questions should help you identify issues that need to be addressed before you start deploying Discovery Server as a strategic component in your IT environment.

- What backup software are you currently using?
- What does your current software license permit you to do?
- Does your backup software support backing up open DB2 and Domino files?
- What backup hardware do you have available?
- What will be the downtime policy for your Discovery Server servers?
- What impact will the scheduled downtime have on each of the Discovery Services (spidering, indexing, and so forth)?
- What backup, recovery, and disaster recovery policies does your company currently have in place?
- How much data do you expect to back up, and at what speed?
- What is the anticipated growth of your data?

9.2.2 The importance of backing up the primary server

It is important to note that regardless of your Discovery Server infrastructure and design, the primary server must always be backed up completely. The new K-map replication features of Discovery Server 2.0 do not allow you to simply replicate the K-map to a secondary server, and then perform the backup via that secondary server.
9.2.3 Online versus offline backups

When considering the backup of a distributed Discovery server environment, it is important to consider the transactional nature of the Discovery server product. Using backup software that supports open files solves the issue of backing up the file system, DB2, and Notes databases that may be in use by a Discovery Server process. However, this only solves half the potential problem.

What about the transactions that are being sent around to the other servers in your Discovery server architecture? For example, you cannot back up the file system-based queue files on your primary server, and then back up the file system-based queues on a secondary server five minutes later. In this example, your backup of the primary server may reference the fact that a spider is still running on a secondary server, but by the time the backup of the secondary server is finished, the spider has completed and sent its completion message back to the primary server. In this case, your restored system could be out of sync, and result in some corruption or odd behavior.

However, if the primary server is backed up, then you will have a full backup of the K-map tables, and you would still be able to recover the system (albeit, with additional troubleshooting and reinstall of secondary servers, re-distribution of secondary tasks, and so forth). This would potentially lead to significant additional downtime and effort required to get your environment back up and running in the case of a major failure.

Thus, the time required to recover a system that has been backed up online must be compared to the nightly shutdown of Discovery servers to get a full system backup, knowing that this nightly downtime will allow for quicker system recovery in the case of a failure.

9.2.4 Establishing a backup schedule

For many system administrators, the most appropriate strategy for backing up Lotus Discovery Servers will not become apparent until some experience has been gained on how the servers are performing and whether they have reached a steady state of operation. A steady state of operation is achieved when a known set of data repositories are being incrementally spidered and no major changes are being made to the volume of data being processed each cycle.

To acquire a good understanding of how your server infrastructure is operating, it is strongly recommended that you implement performance logging of your Lotus Discovery Servers soon after they are deployed.

When planning how to integrate backup of your Lotus Discovery Servers into your company’s backup policy, there are some features of a Lotus Discovery Server which need to be taken into account. In addition, there are issues which
relate to how you have configured your Lotus Discovery Servers and what loads they are subjected to. For example, if you have decided to schedule execution of resource-intensive services for overnight processing, you might find the available window for backing up your primary Lotus Discovery Servers (namely, while the server is idle) is very limited.

Features to consider about your Lotus Discovery Servers are:

- Many of the Discovery Server services are very resource-intensive, and can take a long time to complete processing of large, newly spidered data repositories.
- The Lotus Discovery Server and DB2 services will need to be shut down (or temporary locks will have to be placed on objects to be backed up) if your backup software is unable to back up open files. This downtime will reduce the time available for scheduling of overnight server tasks or, in the worst case, run over into prime time when users expect access to the system.
- Network traffic between the primary and secondary Discovery Server servers can be extremely high. If you are considering a strategy which includes backing up files across the network, make sure you take into account your network bandwidth capacity.

Keep in mind the difference between resource requirements during initial server setup and ongoing resource usage after your system has reached steady state operation. Also consider the impact of spidering large new data repositories. First time spidering may take longer than expected and can generate significant load on your Lotus Discovery Servers and network.

**Critical backup points for the Discovery Server K-map**

Your key backup points will change as you progress down the path of deploying your Discovery Server infrastructure. However, during the initial creation and building of the K-map, there are several key points at which you should back up your Discovery servers:

- After creating the K-map from a representative sampling of repositories, but before K-map editors relabel.
  
  This allows you to recover to the initial machine-generated state of the K-map. Relabelling of machine-generated categories can be a demanding task for the uninitiated and inexperienced K-map editor. If the relabelling process produces unsatisfactory results, the original K-map can be recovered to a known consistent state and the K-map editors can apply what they have learned from their previous efforts.

- After K-map editors have relabeled but before enabling Categorization.
If the categorization process produces unsatisfactory results, the K-map editors can recover their work-to-date and make adjustments to produce a better result.

- At least once a week, depending on the frequency of changes and your organization's backup policy.

### 9.2.5 Choosing a third-party backup solution

We highly recommend that a third-party vendor backup solution be utilized to assist in the backup of your Discovery Server environment. There are a number of third-party software companies that offer backup software for DB2 and Domino, since Discovery Server uses the Domino engine as part of its underlying system architecture. These products are written to appropriate the DB2 or Domino APIs, allowing for the backup of open files, and thus minimizing the downtime of your Discovery Server environment.

The choice of backup products should be made based on your overall computing infrastructure. The following section includes a few guidelines to consider when selecting a backup solution for your Discovery Server environment.

#### Features to look for in your backup software

As part of the process of selecting software for backing up your Lotus Discovery Servers, you should consider the following issues:

- Can the backup software handle both Lotus Domino and IBM DB2 data files?

- A distributed Lotus Discovery Server implementation can generate significant traffic between servers, especially during the initial setup, configuration, and spidering of new data repositories. Will you be backing up across the network, and if so, do you have sufficient bandwidth to handle the load?

- First-time spidering of data repositories can run for a long time (several days in extreme cases). Will the backup software yield if it detects active processes are still running on the server (since you don’t want to lose work processed so far by shutting down or stopping the Lotus Discovery Server)?

- Will the backup software handle problems like file locking, open files, time-outs, and so forth?

- Can the status of existing server processes be checked by the backup software prior to performing scheduled backups? (Running backups concurrently with Discovery Services will significantly degrade server performance).

- The primary Lotus Discovery server can quickly consume large amounts of disk space. How much data do you expect to back up and how long will it take?
What programming interfaces are available with the backup software?

Can the backup software be programmed to allow for special shutdown and restart requirements of a distributed Lotus Discovery Server implementation, should restarts be required? (For example, the primary server is shut down last and restarted first.)

Does the software required dedicated backup media? (Backup media costs can be substantial if you deploy a dedicated tape unit per Lotus Discovery Server).

How easy is it to install, configure, and operate the backup software?

How much training and rewriting of backup procedures will be required?

What is the throughput capacity of the software and do you have the necessary infrastructure to accommodate this?

### 9.2.6 Detailed instructions for performing Discovery Server backups

The product documentation, specifically the Lotus Discovery Server Control Center Guide (ds_admin.chm), provides detailed instructions for performing a backup of a Discovery Server. Review this documentation carefully when planning your Discovery Server backup strategy.

Specific topics included within the documentation are:

- A table defining the key Discovery Server folders and files that must be backed up, divided by the key Discovery services that may reside on a given server. This table is will help you determine which files are required to be backed up on each server in your Discovery Server environment.

  This information is important if you decide to perform selective rather than full backups of each server. For example, a secondary server that hosts only the K-map Indexing Service will only need to back up the files shown within this table for the Indexing Service.

- Instructions on manually backing up the Discovery Server DB2 database tables via the DB2 Control Center.

- Instructions on manually restoring data to a Discovery Server that has been previously backed up via the provided procedures.

- Instructions on manually performing a full, mirror-image backup of a Discovery Server.
9.3 Relocation of Discovery Servers

Often there is a need to change the way Discovery Server servers are deployed. This might mean changing the hardware on which a given server is deployed, or adding more machines to a server cluster.

The following sections provide some manual steps and guidelines for moving your Discovery Server install from one server to another.

**Important:** Try following these procedures on a test environment prior to implementing them on your production environment. This will ensure that you clearly understand all of the steps involved, and that you have caught any additional steps that may be required due to special configurations in your environment.

9.3.1 Moving Discovery Server tasks

If your backend data-repository environment changes, or the load and requirements on your Discovery Server system change, it might be necessary for you to move tasks to a new or different server machine. However, not all services can run on multiple servers, or can be moved to another server:

Services that run on multiple servers, and can be moved to other servers, are the following:

- Notes Spider
- File system Spider
- Exchange Spider
- XML Spider
- Web Spider
- Profile Source Spider
- Profile Synchronization
- Metrics Collection
- K-map Replication

Services that can run on only one server but can be moved to another server:

- Profile Maintenance
- Metrics Processing
- Affinity Processing
- Metrics Reporting

Services that can, once enabled, run on only one specific server:

- K-map Building (ATG)
- K-map Indexing
9.3.2 Changing Discovery Server hardware

There will be occasions when you need to change the hardware used by Discovery Server. This might be a change of disk, adding large disk capacity, or changing to a complete new machine.

In these cases, you need to move the Discovery Server installation carefully. This section explains how to do this.

**Attention:** Movement of a Discovery Server installation can only be carried out if the Windows machine name and DNS hostname remain the same. When you move a Discovery Server installation from one machine to another, you must use the same machine name on the new machine as used on the old machine. Complete steps 1 through 3, then disconnect the old machine and connect the new machine to the network, and proceed with the subsequent steps.

Steps to move a Discovery Server installation:

1. Shut down all Discovery Server servers in the Discovery Server cluster, starting with all secondary servers, and finishing with the primary server.

   Use the `tell ds shutdown` command on the server console of each Discovery server.

2. Shut down all DB2 services by using the Windows Services window to stop all services which start with the characters `DB2`.

   On the Discovery Server primary servers, this will include six DB2 tasks; on Discovery Server secondary servers it will be one or two, depending on the Discovery Server services you have deployed on that server.

3. Perform a backup of your Discovery Server installation.

   Back up the following directories, preserving the directory structure. For example, if you are using a ZIP product, use the `Store full pathname` option or equivalent when creating ZIP file.

   - The Discovery Server program directory (default DS) and all subdirectories
   - The DB2 directory (default DB2) and all subdirectories
   - The DB2CTLSV (default DB2CTLSV) and all subdirectories.

   If you have split the Discovery Server installation over multiple drives, you must create separate backups for each drive.

4. Do a clean install of Discovery Server on the new machine or drive. You must complete the entire install up to the screen "You have completed your Lotus Discovery Server Setup." Do not start the Discovery server after this step.
This step is done to ensure that the appropriate Windows registry entries are created prior to your data restore.

**Important:** You must use exactly the same drives, directories, names, and so forth that were used for the original install.

5. Delete the following directories from the new install. They will be recreated in step 6:
   - The Discovery Server program directory (default DS) and all subdirectories
   - The DB2 directory (default DB2) and all subdirectories
   - The DB2CTLSV (default DB2CTLSV) and all subdirectories

6. Restore from the backup files created in step 3.

You now have a complete Discovery server which matches the configuration that was in place before you moved the installation.
Additional material

This redbook refers to additional material that can be downloaded from the Internet as described below.

Locating the Web material

The Web material associated with this redbook is available in softcopy on the Internet from the IBM Redbooks Web server. Point your Web browser to:

ftp://www.redbooks.ibm.com/redbooks/SG246575

Alternatively, you can go to the IBM Redbooks Web site at:

ibm.com/redbooks

Select the Additional materials and open the directory that corresponds with the redbook form number, SG246575
Using the Web material

The additional Web material that accompanies this redbook includes the following files:

<table>
<thead>
<tr>
<th>File name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>lds2-redbook.zip</td>
<td>Contains the disk space estimator spreadsheet, as described in, “Disk storage requirements” on page 94</td>
</tr>
</tbody>
</table>

System requirements for downloading the Web material

You must have Lotus 1-2-3, Microsoft Excel, or another spreadsheet program with the capabilities to open “.123” spreadsheets.

How to use the Web material

1. Create a subdirectory (folder) on your workstation, and unzip the contents of the Web material zip file into this folder.
2. Open the “LDS Disk Estimator Spreadsheet.123” within either Lotus 1-2-3 or Microsoft Excel.
3. Populate the appropriate values within the spreadsheet for your environment to calculate rough Discovery Server database sizes.
Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the various topics covered in this redbook.

Related IBM Redbooks

- *Inside the Lotus Discovery Server (version 1.1)*, SG24-6252
- *Working with the Sametime Client Toolkits*, SG24-6666
- *Domino and WebSphere Together Second Edition*, SG24-5955
- *Enterprise Business Portals with IBM Tivoli Access Manager*, SGW24-6556

For information on ordering these publications, see “How to get IBM Redbooks” on page 298.

Product documentation

All of the product documentation is available on your Discovery Server after you have completed and installation, or can be downloaded from the Lotus Developer Domain website:


- Discovery Server Control Center Guide (ds_admin.chm)
  Contains the latest install and setup instructions and explains how to use the administrative Control Center to configure and maintain your Discovery Server implementation.
- Discovery Server Readme Help (readme.chm)
  Contains updated release notes for administrators that are more current than those included in the delivered Discovery Server product.
- Knowledge Map Editor Help (kmap_editor.chm)
  For users of the Discovery Server 2.0 Knowledge Map Editor (K-map Editor). Describes how to use K-map Editor to customize and improve a Knowledge Map.
Knowledge Map User Readme Help (KmapUserReadme.txt)
For end users of the Discovery Server Knowledge Map only. Contains release notes on browser issues and K-map access that end users should be made aware of. This is not the comprehensive release notes for the full Discovery Server product.

Knowledge Map Help (K-map.chm)
For end users of the Discovery Server 2.0 Knowledge Map (K-map). Describes how to use the K-map to browse and search for information and work with people.

Referenced web sites
These Web sites are also relevant as further information sources:

- Discovery Server Discussion Forum
  Also available at the Lotus Developers Domain, this is a discussion forum related specifically to the Lotus Discovery Server product.

- Lotus Developers Domain Today
  Read additional technical articles about Lotus Discovery Server in this online technical journal.

How to get IBM Redbooks
You can order hardcopy Redbooks, as well as view, download, or search for Redbooks at the following Web site:

ibm.com/redbooks

You can also download additional materials (code samples or diskette/CD-ROM images) from that site.

IBM Redbooks collections
Redbooks are also available on CD-ROMs. Click the CD-ROMs button on the Redbooks Web site for information about all the CD-ROMs offered, as well as updates and formats.
Index

Symbols
$UpdatedBy 186

A
accented character 127
Affinities 2, 9, 185, 192
   Affinity/Designators group 194
   and privacy considerations 199
   and use of e-mail in generating 198
   changes 200
   control over 10
   designated and declared 194
   how their calculated 192
   publication 193
   user control of 10
   viewing with the K-map editor 11
   what are they? 9
AffinityCalc 38, 41
AffinityProcessing 38
   problems 144
an alternative API 233
   K-map servlet 233
API 2, 14, 231
   getting started 232
   samples 231
Application Programming Interface
   See API
ATG 42
ATGCatToMetricsClassifyQ 20
ATGOutOfBandClassifyQ 19
ATGRoot DocID 44
ATGRoot DocInfo 44
Atomica 260
   installation 264
   Automatic Taxonomy Generator 42

B
Backups 285
   Online versus offline 287
   schedule 287
   strategy 286
Browser-related issues 147

C
Categorization 179
Collaborative Components API 247
   common name 134
Complementary products 244
Completion Queue 22
CompletionQ 19
ConflictHandler 125
conflicts of interest 269
connectivity problems 136
CORBA 53, 86
CPU performance 93
   crashes
      capturing information 276
      types of 271
Cross-certification 154
Custom reports 201

D
Data normalization 42
database.dtd 221
DB2 95
DBCS 127
Directory integration 213
   other considerations 217
DisablePagingExecutive 95
Discovery Server DTDs 220
Disk
   configuration 94
   storage requirements 94
Document Clustering 44
document fit value 40
Document Metrics
   See Metrics
document value 39
document.dtd 221
Domain Search 11
Domino Directory 213
Domino.Doc 255
Domino.Doc spider 30
replication considerations 82
searching 46
valid search operators 48
viewing affinities 11
K-map editor 71
access problems 140
architecture 53
busy message 173
category link 177
communication sequence 53
editing best practices 175
features 171
request retrain 173
working with 170
KmapServlet 15
KmapServlet_AddSearchWildcards 49
Knowledge Management 3
key technologies 4
Knowledge Map 159

L
Label_InXight_tok 44
LargeSystemCache 96
LDAP 213
fields required for Profile source spider 33
Idapquery 15
LDS_CURSOR_DISK 47
LDS_CURSOR_MEMORY 47
LDS_CURSOR_TIME 47
LDSATGCmapWorkQ 19
LDSFullTextWorkQ 20, 50
LDSMetricsWorkQ 19
LDSProfileSyncWorkQ 20
LDSTaxonomyWorkQ 20, 42
Lightweight Third-Party Authentication 214
Locally defined user account 118
Lotus Discovery Server
architecture 13
backend components 16
benefits 6
console 278
deployment 59
external view 14
Installation 107
Interface Services 14
key system files 282
Overview 7
server processes 276

what is it 9
what it does 5
Lotus Discovery Server API Toolkit
See API
Lotus Discovery Server Everyplace 255
Lotus Sametime
See Sametime
LTPA 214

M
Metadata 2
importance of 62
Metrics 2, 37, 186
action constants 187
Affinities
See Affinities
backup 204
collection 10, 38, 186
document constants 187
how they are calculated 190
how they are collected 10
how they are used 189
key field mappings used by 187
metric values 186
Metrics Calculations 186
MetricsCollection 39
reporting 200
restore 204
what are they 9
Metrics Data Flow Diagram 38
Metrics evaluator 70
Metrics processing 38
MetricsWorkQueue 22
Microsoft Exchange spider 32
access considerations 32
Moving Discovery Server tasks 291
Multi-language support 51
performance tuning 51

N
Network impacts 87
Network interface cards 93
normalizer 43
Notes e-mail spider 31
Notes spider 29
field mapping 29
performance considerations 30
security considerations 30
NotesSpiderQ 20
nserversetup.exe 120

P
People 4
People Metrics
See Affinities
people.nsf 41
Performance
Discovery Server specific settings 105
monitoring 100
OS tuning 95
Pilot
preparing for 67
Places 4
Planning 59
assembling the solution team 69
creating a project plan 66
cultural considerations 62–63
identifying the problem 60
knowledge audit 61
strategies for success 74
portlets 246
Primary server 80
replication 81
Profile source spider 33
required LDAP fields 33
Profile synchronization 41
ProfileMaintenance 41
Profiles
issues 144
ProfileSpiderQ 20

Q
QueueCryptoLevel 92
Queues 19
Quickplace 254
Quickplace spider 31
security considerations 31

R
RAM 93
RDSNSServlet 15
readme.chm 108
Redbooks Web site 298
Contact us xiii
Relocation 291
resource usage of tasks 81
reverse proxy 215

S
Sametime 249
and dsmain.nsf 56
configuring integration with 252
DSPAC applet 56
integration 56
people awareness 137
secrets database 138
stauths.nsf 56
STLinks 254
SametimeServer 253
Sample deployment architecture 88
Scheduler 17
handling of spider events 19
Secondary server(s) 80, 128
Security 90
client-to-server 92
data security 92
display LDS ensures 12
repository ACLs and spiders 36
server-to-server 91
user privacy 12
Server administrator 72
Server recovery 268
Server Recovery Plan 268
server.dtd 221
Service Level Agreements 271
Setup 121
common error messages 122
errors that need to be corrected 124
fatal error message 125
single sign-on 214, 216
SLAs 271
Spiders 2
network impact 83
optimal distribution 82
where spider information is stored 22
Sponsor 69
SSL 92
SSO 214
Standard reports 200
stemmer 43
Stopword List 169
stopwords.txt 169
summarizer 44
system files 282

T
tagger 43
Taxonomy 3, 157
    and categorization 179
    choosing repositories 161
    definition and history 158
    field maps 168
    initial edit 174
    key steps to editing via the K-map editor 178
    planning 160
    repository strengths 161
    stopword list 169
    training set 161
Taxonomy Import 181
TaxonomyQueue 22
Team
    assembling 67
Thesaurus 51
    naddenda utilities 52
    service provider interface 52
Things 4
Tivoli Access Manager for e-Business 215
tokenizer 43
Tokens 43
Top configuration problems 132
troubleshooting 268

U
Unable to execute command 125
Uninstalling 130
    if unInstallShield fails 131
Upgrading 110
    from a beta build 115
    manual steps 110
    when to upgrade 109
User profile
    what is it 9

V
VPN 87

W
Warning for incorrect user account 117
Warning for incorrect user account 117
Web spider 23
    authentication 26
    child threads 24
    control instance 23
    metadata considerations 25
    performance considerations 26
    security considerations 26
WebSpiderQ 20
Win32PrioritySeparation 96
Windows Task Manager 273

X
XML 218
XML DTD 14
XML spider 34, 217
    how it works 220
    how to use 219
    operation 229
    sample XML files 223
    settings 230
XMLSpiderQ 20

Z
zip files 161
Lotus Discovery Server 2.0
Deployment, Planning and Integration

The Lotus Discovery Server is a knowledge server. It provides search and expertise location solutions designed to ensure that the relevant knowledge and collective experience of an organization is readily available to help individuals and teams solve everyday business problems.

In this IBM Redbook we provide an in-depth discussion of the architecture and capabilities of the newest release of the Discovery Server - Discovery Server 2.0. We discuss the key aspects involved in planning for the deployment of this product, from both organizational and technical viewpoints.

We also discuss: the key elements of a solid infrastructure design to ensure scalability and reliability, development and maintenance of an enterprise taxonomy, integration of Discovery Server into your existing environment and applications, usage of the Discovery Server API toolkit, and ongoing Discovery Server maintenance.

This book is appropriate for technologists and project managers who are involved with Lotus Discovery Server-related projects.