8th International Workshop on High Performance Transaction Systems, Asilomar, September 1999

Evolution of Groupware for TP/Business Applications: Lotus Domino/Notes

C. MOHAN

mohan@almaden.ibm.com
www.almaden.ibm.com/u/mohan/

IBM Almaden Research Center
650 Harry Road, K01/B1
San Jose, CA 95120, USA

Agenda

- Introduction
- Semi-Structured Data Management
- Storage Architecture
- Replication
- High Availability
- Log-based Recovery
- Summary

☛ www.lotus.com, www.notes.net, www.iris.com, www.edge.lotus.com

Introduction: Groupware

- Originally intended for use in a work group environment
- More than merely messaging
- Lotus Notes established the area in 1989
- Subsequently, Novell GroupWise, Microsoft Exchange
- More recently, many web-based products/services, including realtime collaboration (e.g., instant messaging)

Introduction: Lotus Notes

- **Groupware** product based on Plato Notes (1973) at UIUC
  Runs on numerous systems (NT, Unix, S/390, AS/400, Mac, ...)
- Core development from 1984 by Iris Associates, now subsidiary of Lotus
- **12/89:** Notes 1.0 ships with email, phone book, discussion DB, document DB and RSA public key
  World's first semi-structured DBMS!!
- New terminology since R4.5
  Domino = server   **Notes** = client
  DB functionality almost same in both
- Latest release: **R5.0 (4/99)**
- Sample customers and apps
  - Chase Manhattan Bank: Relationship Management System
  - Bank America: Loan syndication application
  - ABSA: ATM crime tracking system
  - Banc One Financial Card Services: Customer Support System
  - EDS, IBM: Enterprise-wide messaging and discussion DBs

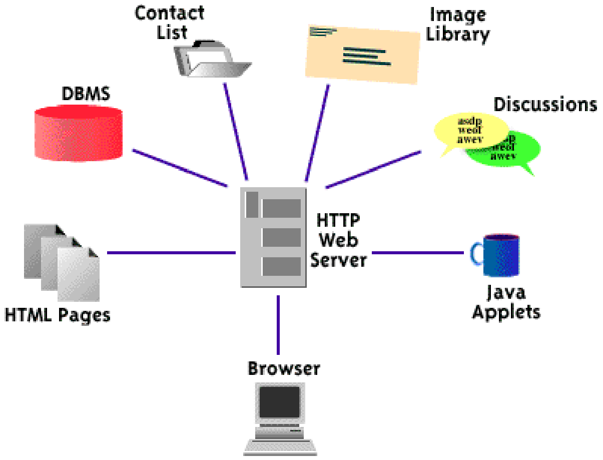
Lotus.
An IBM Company

iris
ASSOCIATES

R5

Functionality

- Messaging
- Discussion DBs
- Calendaring & Scheduling
- Replication
- Agents/Workflow
- Search/Index
- Content Management
- External Data Access
- Fine-Grain Security



➤ Seamless development for multiple client types (Notes, browser)

Functionality Evolution

➤ Transactions and high scalability in R5

Major Advances in Web Development

SMTP, IMAP, MAPI, POP, LDAP	HTTP	NNTP	IIOP	NRPC

Best of Breed Internet Messaging

New Levels of Scalability, Reliability, Performance

OBJECT SERVICES API (OLE, Java, LS, C++, C)				
ROUTING WORKFLOW	VIEW INDEXING	FULL TEXT INDEXING	REPLICATION	ENCRYPTION DIGITAL SIGNATURES
ACCESS CONTROL	DIRECTORY	DOCUMENT CONTAINER	CALENDARING	SCRIPT ENGINES
OBJECT SERVICES				
OBJECT STORE				

Continued Innovation in Collaboration and Knowledge Mgmt

Increased Openness, Ease of Administration and Usability

➤ 34 million seats (YE98)

Many invaluable features for being a web server

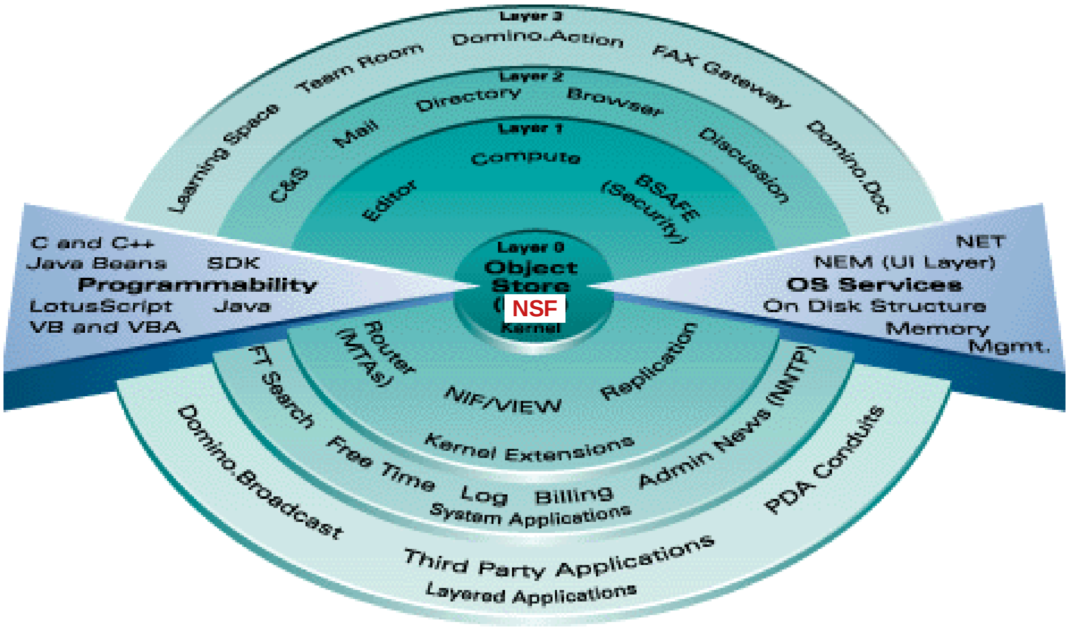
Domino and Transaction Systems

Complementary Functionality

- Documents
- Semi/unstructured data
- Distributed
- Disconnected & online
- Workgroup-oriented
- Ad hoc workflow processes

- Transactional
- Structured data
- Centralized
- Online
- Enterprise-wide
- Structured business processes

Layered Architecture



Database Characteristics

- Schema less, forms are only guides
- DB contains semi-structured documents (docs)
- Docs may be addressed by key, by name or via views
- Views are materialized using lazy update mechanisms
- Heavy use of timestamps to determine changes for view update, replication, etc.
- Objects (e.g., OLE objects): free-standing or doc attachments

Storage Architecture

Database Title
Replication Settings
Access Control List
Replication History
User Activity Log
Policy Note
Help Note
View Note
Form Note
Filter Note
Data Note (document)
Data Note (document)
Form Note
View Note
Data Note (document)
. . .
Collections (indexes)

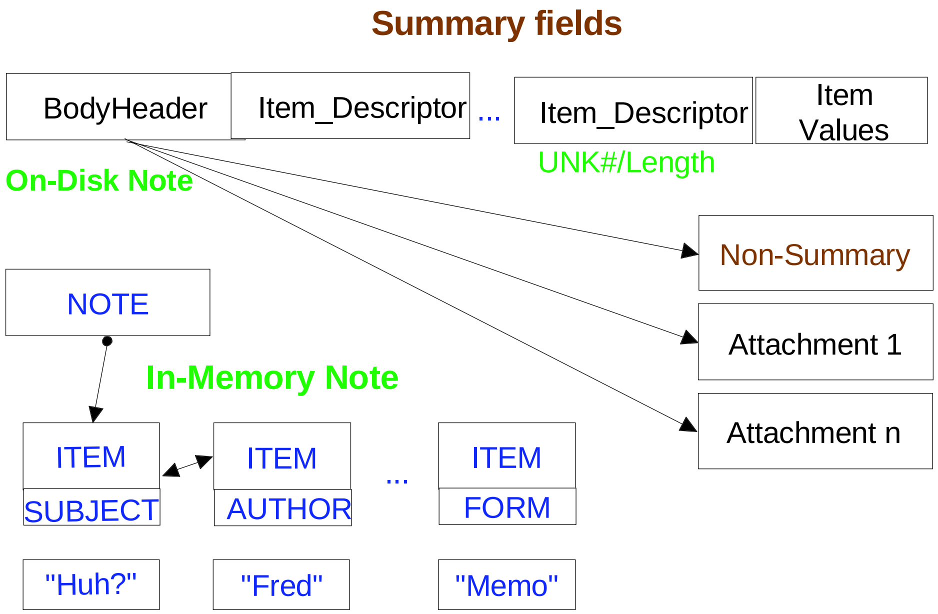
➤ **Structure of a DB**

- A DB = a single self-describing file
- Location independent DB contents
- Most data stored in machine independent form on disc - allows binary file copy across machines (e.g., RISC and PC)
- Docs are also self-describing
- Within DB, separate storage of structured and unstructured (e.g., attachments) fields

Basic Constructs: Notes

- Used to store variety of information - e.g., views, forms, icons, and policy data
- Docs = a kind of note (data note)
- Arbitrary # of items, any item name/type, any # of instances of each
- Split into summary, non-summary, and attachments
- Identified and locatable by NotesIDs (DB specific) and UNIDs (universal)
- Have parent/child relationships (e.g., original and response in a discussion DB)
- Timestamp fields for skimming, precedence determination

Note Structures



Summary fields

BodyHeader

Item_Descriptor ...

Item_Descriptor

Item Values

UNK#/Length

On-Disk Note

Non-Summary

Attachment 1

Attachment n

NOTE

In-Memory Note

ITEM SUBJECT

ITEM AUTHOR

...

ITEM FORM

"Huh?"

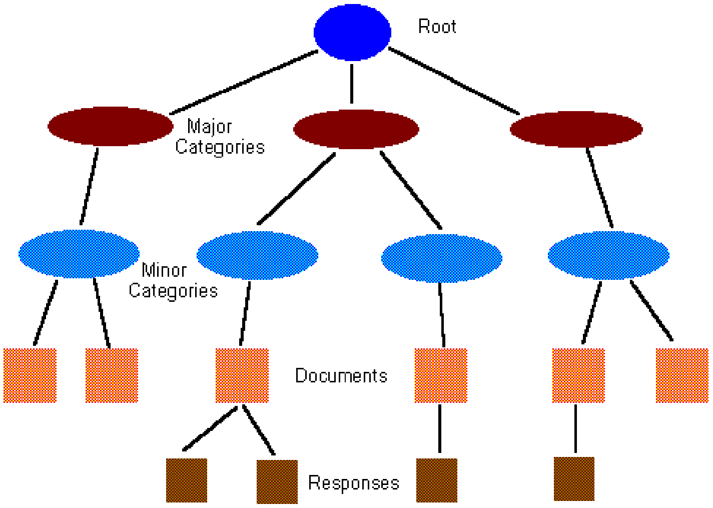
"Fred"

"Memo"

Basic Constructs

- Forms - Templates used to view/update data via GUI
- View Note: Stores rules for
  - Selecting docs from a DB (view definition formulas)
  - Organizing docs
  - Presenting info of each doc
- Collection - An instance of a view at one moment in time (materialized view with periodic, timestamp-based refresh)
  - Always maintains its tree-like structure
  - If doc in 2 categories, collection has 2 nodes for same doc
  - A note's indent level tells if it is a main doc, a response, a response to response, ...
  - Collapsible and expandable sections

Collection Hierarchy



- ➤ List only main-topic docs in category, ignoring response docs
- ➤ Find only responses to a certain main topic
- ➤ Get list of all docs in a collection, regardless of any categories
- ➤ Response doc contains UNID of parent doc (no inverse link)

Views

- Each view stored in a container: Set of 8K pages
- View represented by 2 or more B-trees
- Collation options include:
  - Multi-column ascending/descending
  - Secondary collations
  - Permutations with or without categorization
- B-trees used for collations are hierarchical and ranked

Basic Constructs

- Folders - Very similar to views, except that user defines contents; nothing like view definition formula
- Events
  - Types and severity associated with events
  - System and user-defined events
- Agents: Support for ECA rules
- Mail - Important components of mail include mailer, router, and Name & Address Book
  - Editor invokes mailer when doc should be sent
- Shared mail DB
- Access Control List (ACL)
- Full text indexing (Verity engine): index stored externally
- Add-ins: FT engine, replicator, extended search, ...

Data Types

- Fields - Only summary fields (size limit: 15K total) can be used in a view selection formula or an NSFSearch formula
  - Name, data type, data length, value, flag word
  - 4 *basic* types: text, number, time/date, text list
  - User-defined types: Store any data in field. Notes makes no attempt to interpret it. Via GUI cannot see them from a Notes form or in a Notes view.
- Rich text fields
  - Used to store a variety of objects, including text, tables, document links, bitmaps, and OLE links
  - Rich text item = composite data (CD) records defining item's components (data and metadata)
  - Cursor used to navigate among RT item's elements

Data Access and Manipulation

- Navigator object: Goto{First, Last, Child, Parent, NextSibling, NextParent, NextChild, NextUnread, NextCategory, Previous, LastSibling, ...}, GetDescendents
- Formulas: Input to NSFSearch, View definitions, ...
- Steps to write a doc
  - Opening/creating
  - Writing fields
  - Save
- Creating/opening note causes in-memory object creation
- Any write to a field held in memory until note write to disk

Some NSF API Calls

- NSFNoteCreate - Creates a new, empty in-memory note
- NSFItemAppend - Add a field to note
- NSFNoteUpdate - Write in-memory note to on-disk DB
- NSFNoteOpen - Opens note with specified NoteID
- NSFNoteDelete - Deletes note from DB
- NSFSearch - finds notes of any class via sequential search or DBs in directories/servers
  - Selection formulas consist of same @ functions, field names, logical operators as in view selection
  - For each note found matching selection criteria, specified action routine called to process note
- NSFItemGetText - Read fields of 4 basic data types
- NSFItemInfo - General function to read field of any type
- NSFItemIsPresent - Check if a field is in a document

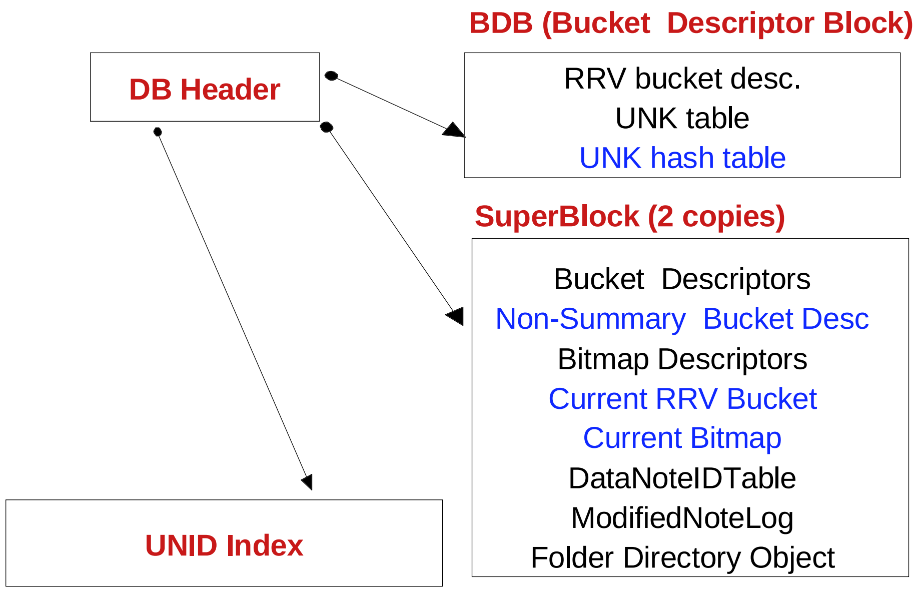
Some NIF API Calls

- NIFFindView - Finds note ID of view, given view name
- NIFOpenCollection - Opens collection of a view
- NIFReadEntries - Scans collection from specified starting position in specified manner, returning specified number of entries
  - A flag to skip to next entry at same level in view hierarchy as current one
  - A flag to go down one level into subcategories
- NIFUpdateCollection - Resynchronizes collection with DB
- NIFFindByName - Search collection for first note with primary sort key matching given ASCII string and return # of notes with this key
- NIFFindByKey - Search collection for first note whose sort keys matches those specified, and returns location of note in collection and also # of notes that match

Architecture

- Like web, based on rich document-oriented DB, fielded forms and document linking
- Original design point: small workgroup
  Led to scalability and data recovery problems with expanded usage and new requirements
- Recent work (R5): Far improved scalability allowing enterprise-wide deployment
- Domino = Classic Notes + security and network management + HTTP and CGI support
  (A fully standard, Java-compliant, web server)
- HTML, Java, SSL, NNTP, IMAP and LDAP are integrated
- URL syntax for addressing Notes objects
- DB functionality in Domino (server) and Notes (client) almost identical

Major Meta-Structures



DB Header

BDB (Bucket Descriptor Block)

RRV bucket desc.
UNK table
UNK hash table

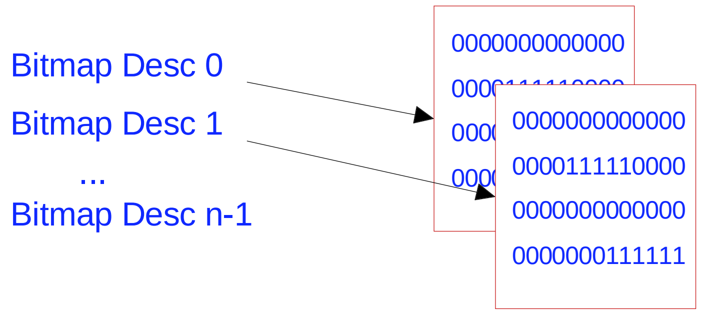
SuperBlock (2 copies)

Bucket Descriptors
Non-Summary Bucket Desc
Bitmap Descriptors
Current RRV Bucket
Current Bitmap
DataNoteIDTable
ModifiedNoteLog
Folder Directory Object

UNID Index

Bitmap Descriptors

Control DB space allocation



Bitmap Desc 0

Bitmap Desc 1

...

Bitmap Desc n-1

0000000000000

0000111110000

0000000000000

0000000111111

Each bitmap controls 2048 x 8 x 256 database bytes

Bitmap n controls DB space starting at n x 2048 x 8 x 256

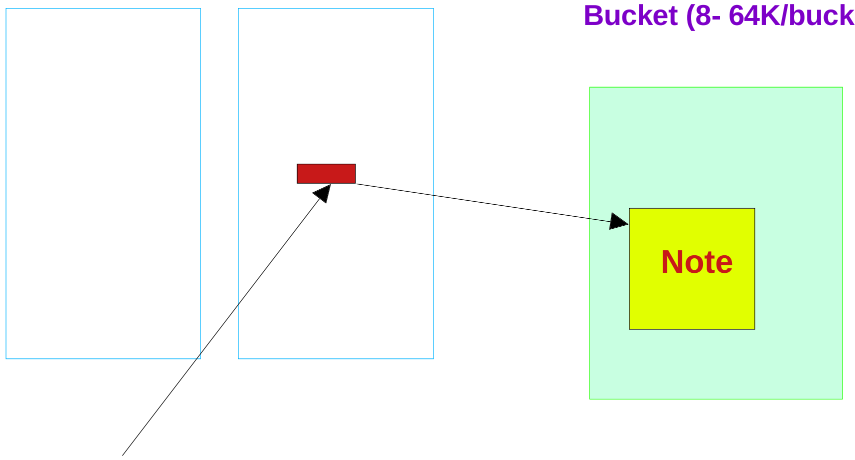
No page headers or trailers

RRV Buckets

Map NoteIDs to note location (4 byte addr in file)

RRV Buckets
(8K/bucket, 8 bytes/entry)

Bucket (8- 64K/bucket)

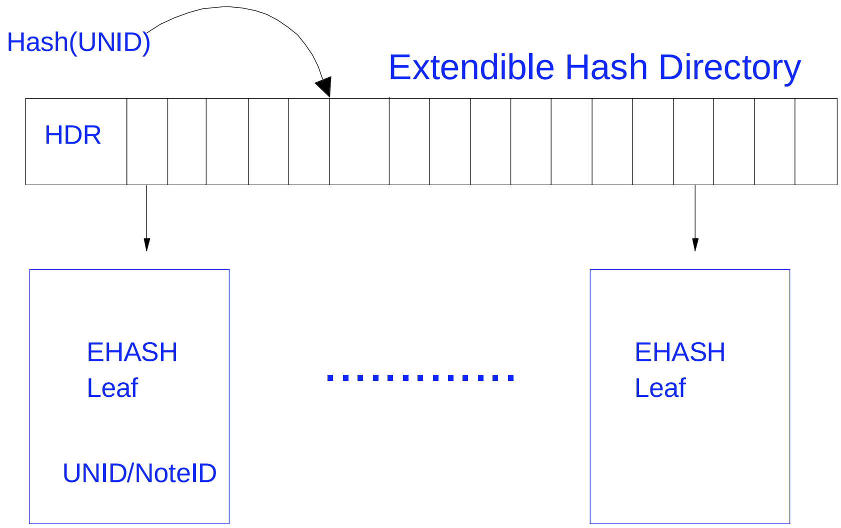


Note

NoteID

NoteIDs are assigned sequentially to help compress NoteID lists

UNID Index



UNID->NoteID index

Buckets

- Storage location for (parts of) documents
- Summary and non-summary types
- Variable sized (4-64K, power of 2)
- Sys R style slotted page structure
- Bucket descriptors contain:
  - Disk position of bucket
  - Bucket modified time
  - Bucket size
  - Bucket free space

I/O

- Buckets read and written as units
- Index pages read and written as units
- Bitmaps, other meta-objects read and written as units
- Other objects read and written as byte ranges - relies on OS file buffering
- File syncs used to force data to disk - for meta operations and on request via API
- File and container level bits to detect need for fixup

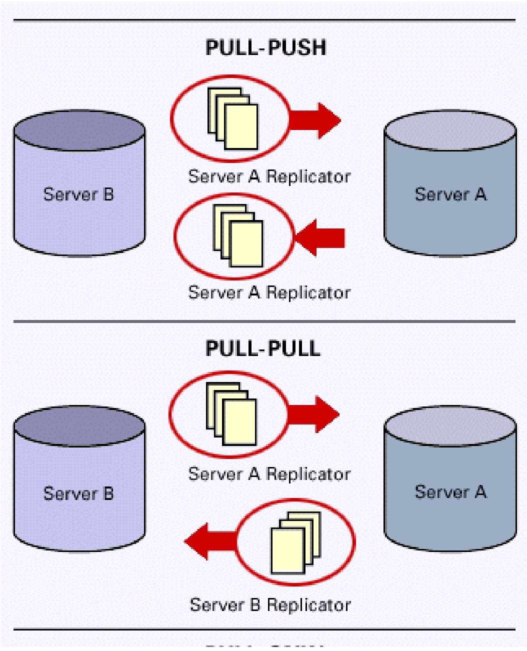
Extensible Architecture

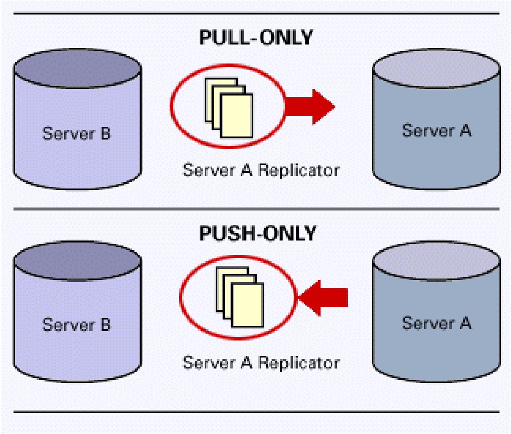
- Extension Manager
  - Allows an executable program library in an app to register callback routine to be called before, after, or before and after Domino/Notes performs selected internal operations
  - Allows API apps to perform database shadowing or add additional access controls
  - By trapping replication conflict notification, can automatically handle replication conflicts

Replication

- Replication can be selective and with any server with a copy
- Replication conflict detection at document or field level
- Push and pull supported
- At form design time, can enable automatic merge of document versions by Replicator if no fields conflict
- During form creation, can request versioning of documents on update
- Code custom conflict handler with LotusScript
- Note modification timestamps used to identify changes
- Deletes replicated by having tombstones (stubs)
- Same ReplicaID in replicas of a DB
- Same UNID in replicas of a note
- Even without replication, conflicting updates possible due to locks not being held after client reads but before it updates

Replication

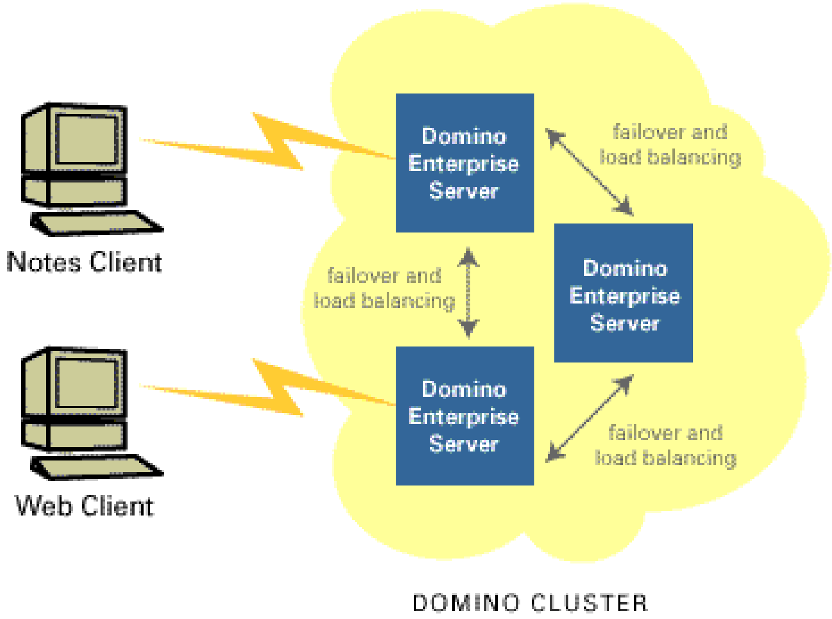




High Availability: Domino Cluster

- Group of 6 Domino servers connected to form a team that cooperates to provide services or resources to clients
- Typically, each cluster has multiple replicas kept tightly synchronized by Cluster Replicator - shared nothing architecture with enhancements to normal replication
- Advantages: high data availability, tightly synchronized databases, scalability
- Provide failover protection for critical DBs and servers
- With failover, users can still access DB on server failure
- Workload balancing feature: Heavily-used servers can pass requests to other cluster servers to evenly distribute work

Domino Internet Cluster Manager



Log-based Recovery

- New feature of R5: Result of joint work between Dominotes project at IBM Almaden and Iris
- Logging optional at DB granularity
- Implicitly each API call treated as an ACID transaction
- Single log per server
- Extensions to ARIES to permit LSN-based recovery, and to handle unlogged updates to attachments, and for switching between logged and unlogged DB modes

Recovery Complications

- Original design of storage management not done with recovery in mind
- Too many persistent structures
- A single file with different data structures: B+-tree, hash access method, bit maps, summary buckets, non-summary buckets, attachments' storage, lists, arrays, ...
  - Some are paginated, others are byte streams
  - Some pages have headers and trailers, others don't
  - Pages are of varying sizes
- Some structures paginated, others just byte-streams
- Structures get allocated, deallocated, migrated
- How to handle situations where user overwrites an existing database file with an older/newer replica?

ARIES Extensions

- Could not afford/tolerate complete redesign of on-disk formats to conform to ARIES requirements
- Use of traditional DBMS as persistent layer also ruled out due to complexity, Notes API semantics + flexible data model, ...

Evolution was needed rather than revolution!!

- Preanalysis of log to identify data structure allocations/deallocations and logging of migrated data
- Deal with direct I/Os that bypass buffer pool and use of OS file caching
- Eliminates "fixup" at restart after failure and allows fuzzy backups

Enterprise Integration

RDBMS, TP and ERP systems working with Domino

- Bi-directional data exchange with transaction integration
- Workflow
- Mail and messaging
- Client Integration

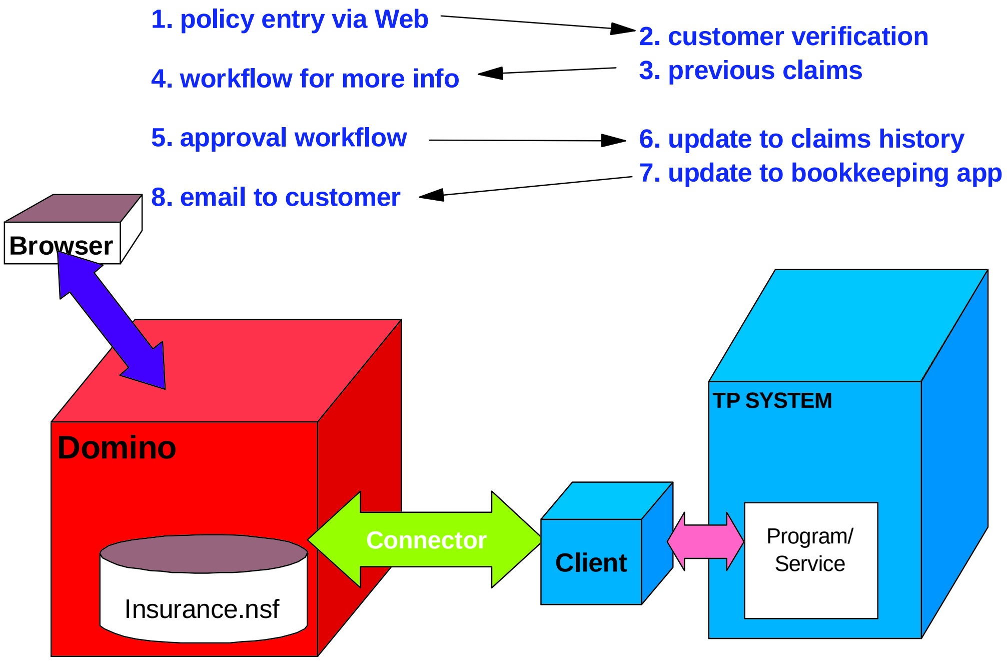
Business Benefits

- Unify disparate backend systems
- Link business process with record keeping
- Flexibility when business changes

IT Benefits

- Separate core users from casual users
- Deliver functionality to additional audiences
- Fastest way to web
- Cheaper to deploy than transaction systems for all

Insurance Policy Claim



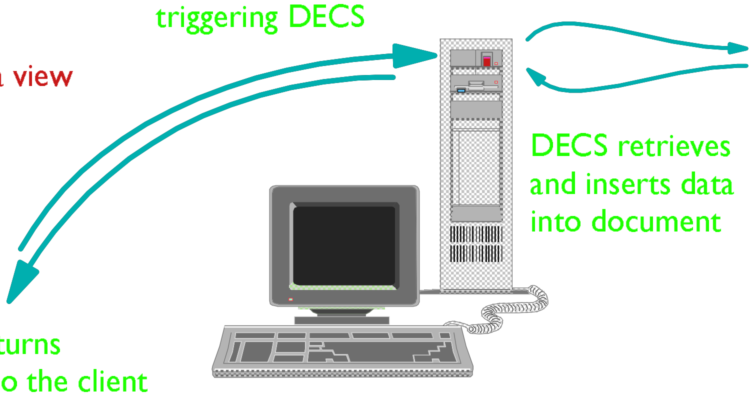
1. policy entry via Web
2. customer verification
3. previous claims
4. workflow for more info
5. approval workflow
6. update to claims history
7. update to bookkeeping app
8. email to customer

Lotus NotesPump

- Optimized for moving large volumes of data
- Replicates dissimilar databases
- Bi-directional updates
- Managed using Lotus Notes client
  - Form based UI
- No API programming required
- Scriptable
  - LotusScript BASIC
  - CGI

DECS: Domino Enterprise Connection Services

- Event driven
  - Document events ...
    - create, open, update, delete



click on document in a view

Client requests document from Domino server

Domino retrieves the document from the NSF triggering DECS

DECS requests data from external server

DECS retrieves and inserts data into document

Domino returns document to the client

Domino Server

Other Topics

- Agreement with
  - AOL to deliver Headlines My News
  - Lycos to deliver Network info to the Notes R5 Welcome page and permit web searches
- Support for Java-based servlets in R5
- LDAP V5
- X.509 V3 certificates and S/MIME for secure Notes to non-Notes email
- Use built-in web server or IIS
- Employ framesets, image maps and Java applets in Notes forms and documents
- Sametime for instant messaging
- For Notes-based collaboration, TeamRoom template: infrastructure for managing deadlines, organizing docs via categories and routing data to project participants
- QuickPlace for web-based collaboration

XML

- Available
  - LotusXSL: Open source Java implementation of XSL Transforms (XML to XML/HTML/SGML)
  - Can be invoked from Domino Java servlets or agents
- In the future
  - Native storage of XML in Domino, providing security and replication
  - Runtime API support for standard XML libraries (DOM Level 1 and SAX - Simple API to XML) with language bindings for the APIs for LotusScript, Java and COM

Domino Workflow

- Functionality coming from acquisition of OneStone with its Prozessware WFMS
- Graphical, high level process definition capabilities
- Rule-based routing, sophisticated role directory, library of reusable components
- Exchange: 3rd party workflow tools on top

Summary

- Mature and very successful 10 year old product
- Has basic features of a semi-structured DBMS
- Limited support for "edges" (inter-doc links)
- Declarative query capabilities need major improvement
- Various tradeoffs involved in using an RDBMS vs Domino for storing semi-structured data
- Better workflow support via OneStone acquisition
- Scalability and recovery vastly improved in R5 due to work at Iris and Almaden (Dominotes project)
  - Work needed to exploit log for index refresh, replication, etc.
  - Locking granularity needs to be reduced and transaction APIs need to be exposed for user-specifiable transaction boundaries