Open Hypermedia

- Sun Link Service
- Microcosm
- Devise Hypermedia

- (Chimera + a few more)

Sun Link Service

- The first attempt in creating an open hypermedia system
- Distributed together with NSE from Sun in a period
- Principle:
  - links are stored outside documents via communication with a link service managing a link-database
    - hypermedia user interface is implemented by the link service
  - a link library to be used during development of new applications or later source-kode tailoring
    - the library implements link-markers and communication to the link service
- Idea didn’t become sufficiently popular among the development groups
  - Sun Link Service is out of distribution
  - (Asimilar service (LinkWorks) is however still being distbuted by DEC)

Sun Link Service: requirements for applications

In addition to the use of the link library:

- ■ management of unique keys (ASK) for application objects
- ■ send ASKs to the link service
- ■ mark an applikation object given an ASK
- ■ validate existence of an application object with a given ASK
- ■ (Use of C as implementation language)

- ■ ASCII text - a full line of text is used as ASK
  - • problematic: vulnerable with respect to changes - too coarse granularity

(ASK are similar to Meyrowitz’s Persistent Selection)

Sun Link Service: worth to notice

- versionning and merge of link databases to support cooperation
- identifies dangling link problem
  - a dangling endpoint is detected with an exception mechanism during follow
  - garbage collection for link databases
-

(Chimera)

- Research prototype from University of California at Irvine (UCI)
- Same idea as Sun Link Service, but more avancered set of hypermedia concepts
  - Use Chimera libraries during development
  - Source-code tailoring
  - Wrapper Viewer client + application macros
    - third party applications cannot themselves become Chimera clients
    - introducerer et en ekstra process
- Supports multiple programmering languages
- Chimera client can have multiple views on the same document/data-object
  - anchors are relative to views instead of documents

Microcosm

- Developed at University of Southampton
  - product version distributed by Multicosm ltd. (www.multicosm.com)
- Integration of real third-party applications
  - minimal requirements for applications
- New link concepts - “Ending the tyranny of the Button”
  - ‘generic’ and ‘local’ links
- Many use experiences
  - education
  - industry (CAD)
  -
- Several platforms
  - primary Windows but also Unix and Mac

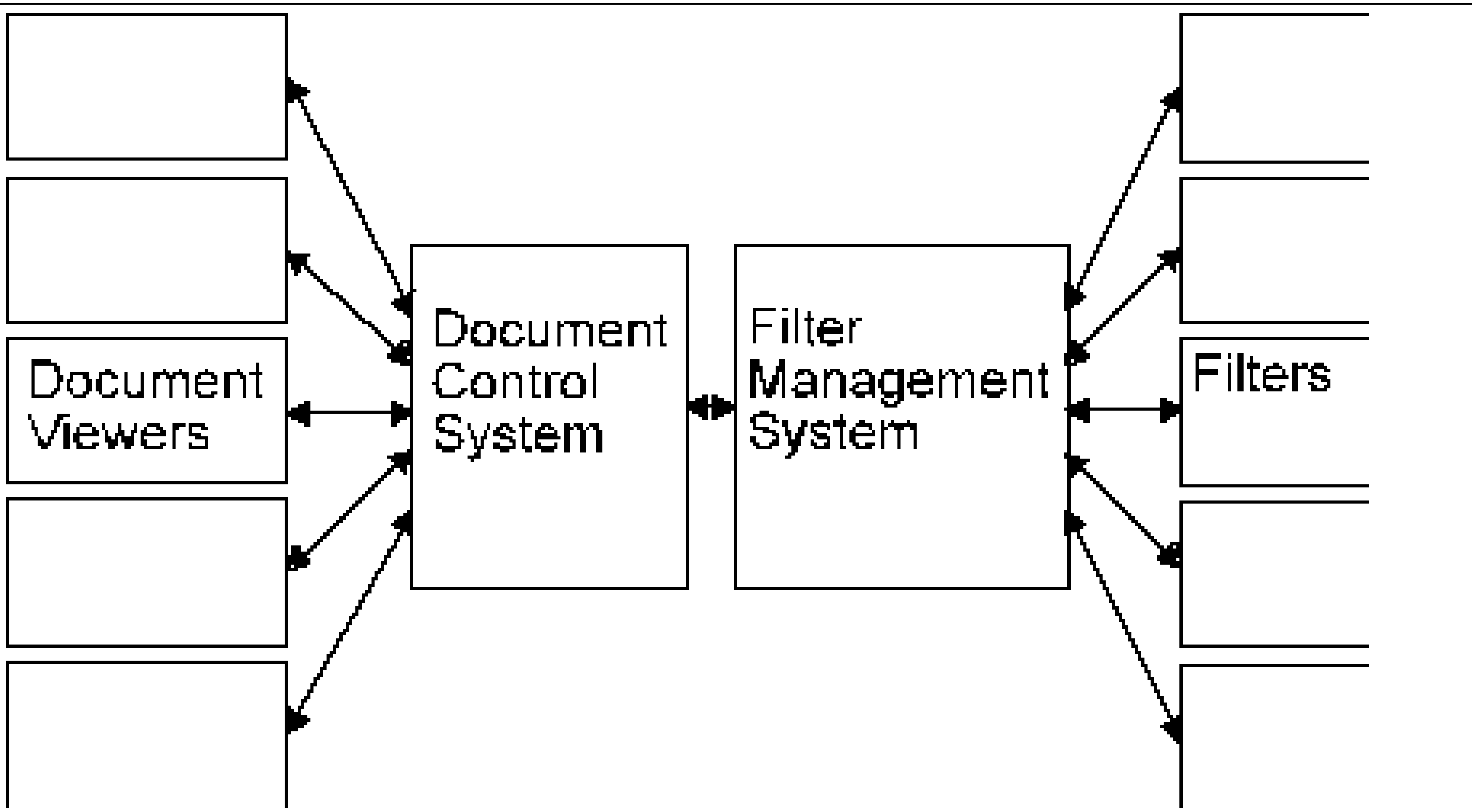
Microcosm: Link concepts

- **Specific link**
  - from an object at a *specific* location in a source document to an object at a specific location in a destination document
- **Local Link**
  - from an object at an *arbitrary* location in a source document to an object at a specific location in a destination document
- **Generic Link**
  - from an object at an *arbitrary* location in an *arbitrary* source document to an object at a specific location in a destination document

("object" is typically a textstring)

- **Text retrieval links - computed destination**
  - search for string match across registered documents
  - lookup in inverted indexes and computation of "similarity coefficient"

Microcosm: Architecture



Microcosm: Integration principle

- No requirements for applications
- Closed applications
  - launch-only for all applications
  - follow link from applications when copy to clipboard is supported
  - Universal Viewer (Parasite-programme)
- Open applications
  - use of macro language or similar to extend the user interface
  - communication of textual messages about links and anchors
- Protocol - simple communication tagged messages
  - messages are interpreted of one or more filters in the chain
- Integration based primarily on Generic Links and Local Links using string match

Microcosm: Assessment

- Very comprehensive experience with integration of third pary applications
- Nice implementation - most comprehensive so far
- Automatic generation of generic links a great advantage
- Universal Viewer - smart trick

But

- very text-oriented - problems with anchoring in graphics and similar
- hypermedia model is very simple: 2-ary links, no composites,..
- lacking multi-user architecture

Devise Hypermedia

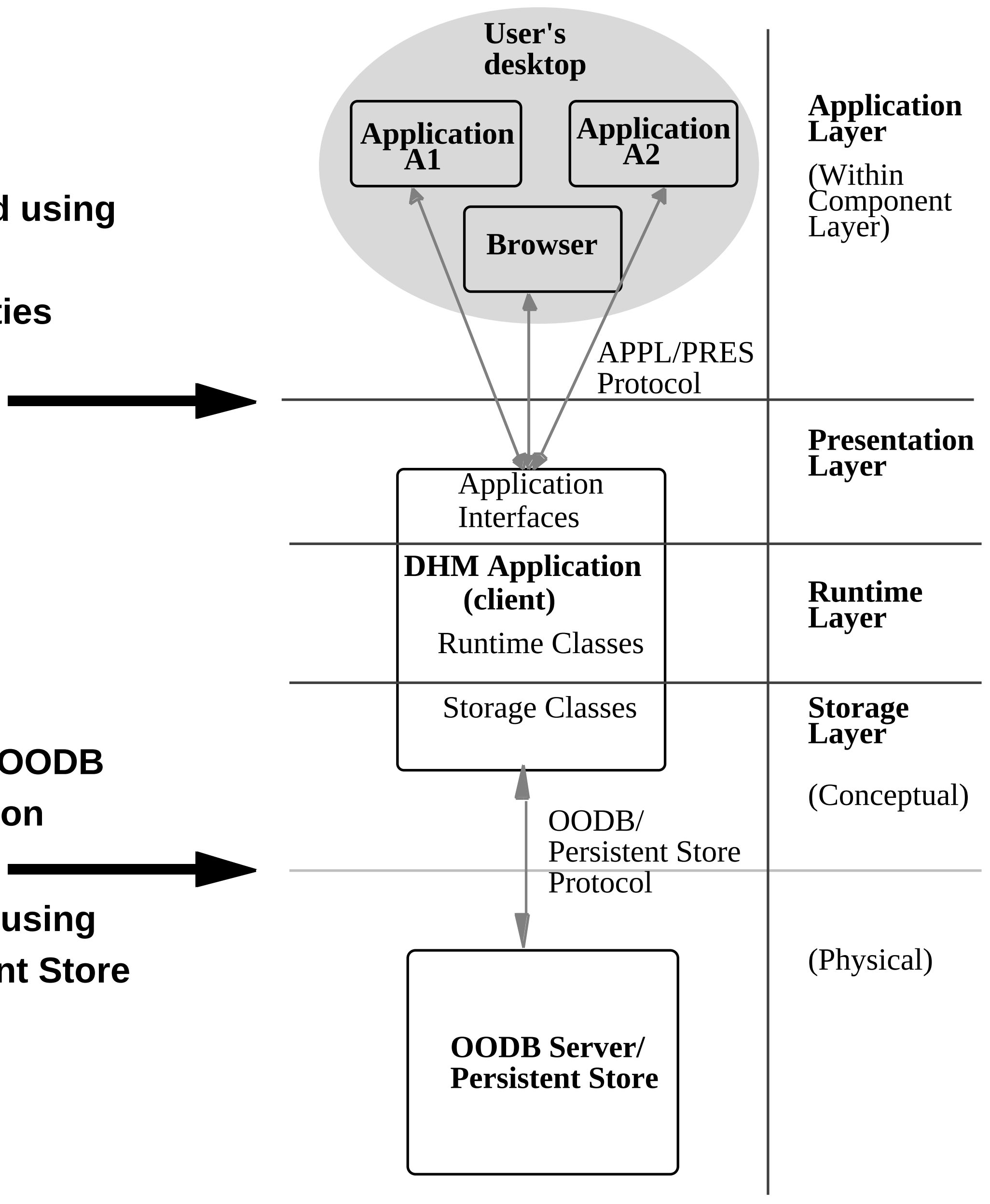
- Industrial prototype developed at DAIMI with EU Esprit support from 91-95
  - product soon
- Supports the generality of the Dexter model
- Based on a tailorable Application Framework
- Multi-platform concept
- Includes multi-user architecture

Devise Hypermedia: Architecture

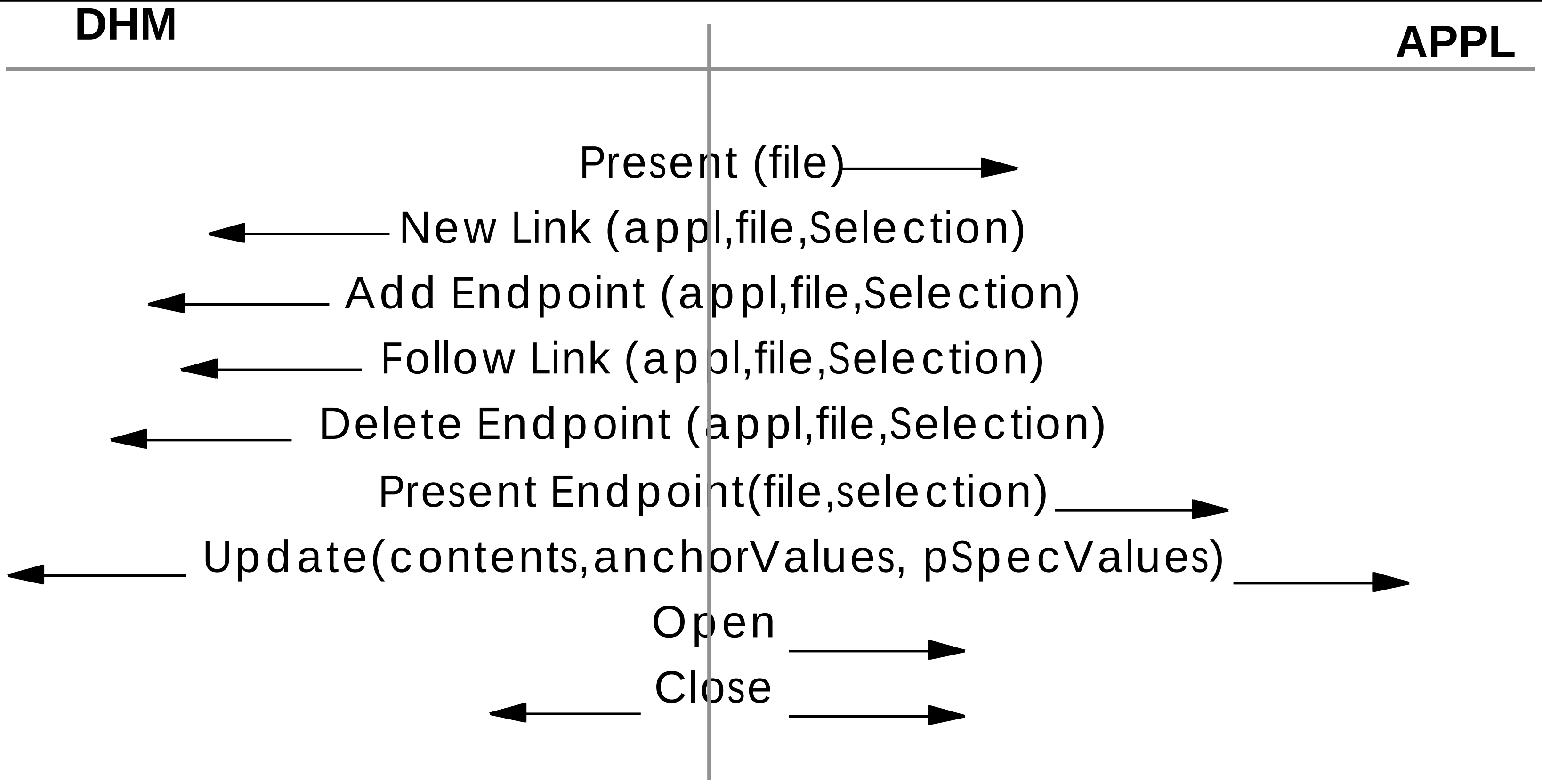
Protocol implemented using platform dependent communication facilities

- DDE/OLE2
- AppleEvent
- Sockets
- Beta distribution

Either
Communication with OODB using BETA distribution
or
Direct procedure call using the included Persistent Store library



Devise Hypermedia: Simple integration protocol


DHM
APPL
Present (file)
New Link (appl,file,Selection)
Add Endpoint (appl,file,Selection)
Follow Link (appl,file,Selection)
Delete Endpoint (appl,file,Selection)
Present Endpoint(file,selection)
Update(contents,anchorValues, pSpecValues)
Open
Close

Example Implementation;

Excel AppleEvent with the following text data:

#XCL#DEVISE1:HYPERMEDIA:DEMO:TEST.XL#NL#1#R1C1:R4C5

Devise Hypermedia: Excerpt from protocol

- *newLink*: 0

  Usage: create an new link with a "source" anchor/endpoint.

  Example: gnuemacs,0,/users/kgronbak/DHTK/docs/interface.tex,1,3

- *newLinkDest*: 1

  Usage: create an new link with a "destination" anchor.

  Example: gnuemacs,1,/users/kgronbak/DHTK/docs/interface.tex,1,4

- *newLinkBoth*: 2

  Usage: create an new link with a "both" anchor.

  Example: gnuemacs,2,/users/kgronbak/DHTK/docs/interface.tex,1,5

- *addEndPoint*: 100

  Usage: add a "destination" anchor/endpoint to the current link.

  Example: gnuemacs,100,/users/kgronbak/DHTK/docs/interface.tex,1,6

- *addEndPointSou*: 101

  Usage: add a "source" anchor to the current link.

  Example: gnuemacs,101,/users/kgronbak/DHTK/docs/interface.tex,1,7

- *addEndPointBoth*: 102

  Usage: add a "both" anchor to the current link.

  Example: gnuemacs,102,/users/kgronbak/DHTK/docs/interface.tex,1,8

Devise Hypermedia: Principles for integration

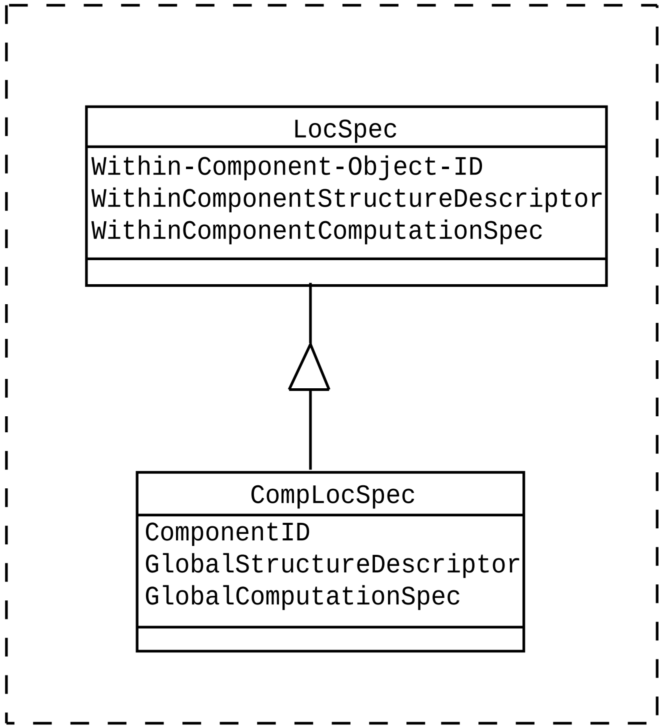
- ■ Whole documents from any application may function as link endpoints (launch-only).
- ■ Links to parts of documents require that the applications are open to communication via DDE, AppleEvent, etc.
- ■ There should be a macro-programming language
  - to extend the user interface
  - to communicate with the hypermedia service application
- ■ The applications' document format remain unmodified
  - built in mechanisms like BookMarks, cell-names, and CAD object ID's are used as anchor values
  - positions may used in write-protected documents.
- ■ The hypermedia client is customized for new applications using object-oriented specialization.
- ■ An exception-handling mechanism is used to handle deletions of documents and of portions of documents containing links.

Levels of application openness

- Closed applications
  - May be handled with Universal viewer
- Applications supporting communication about selections, but no tailoring
  - Not possible to add menues and macros within application (e.g. Excel 4)
- Applications supporting communication about selections and tailoring
  - Example: Emacs
- Applications supporting communication, tailoring, and persistent selections
  - Examples: Word and excel
- Applications supporting object distribution, tailoring, and persistent selections
  - Examples are fully OLE and OpenDoc supporting applications

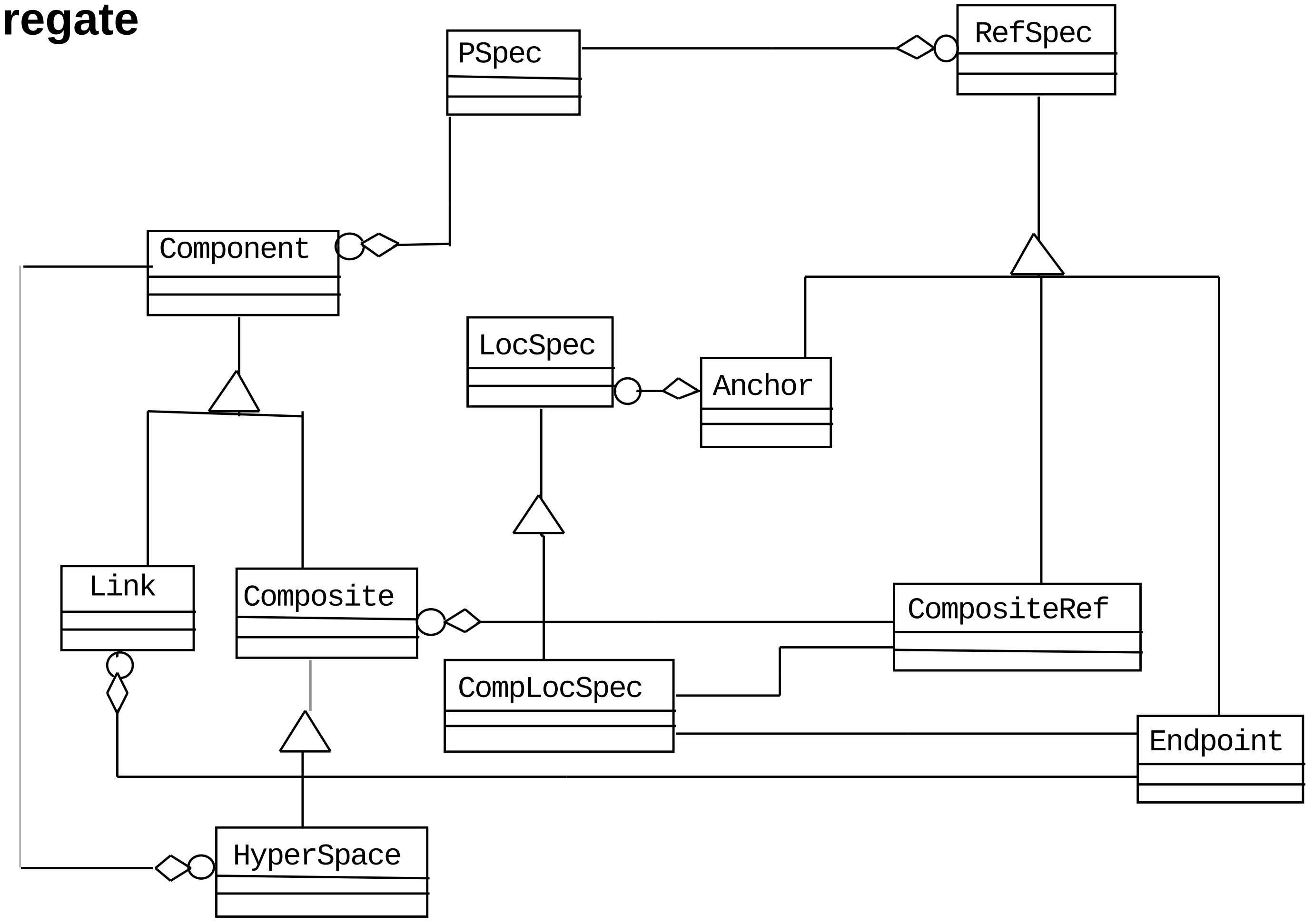
Open Dexter framework: Location Specifier classes

- Cross component vs. within-component location
- Support both computed and static locations
- Redundant information for handling link inconsistencies


LocSpec
Within-Component-Object-ID
WithinComponentStructureDescriptor
WithinComponentComputationSpec
CompLocSpec
ComponentID
GlobalStructureDescriptor
GlobalComputationSpec

Open Dexter framework

- Anchors aggregate LocSpecs



LocSpecs for different media

- Text: (Locating spans of text)
  - Object ID: a bookmark ID
  - Structure Descriptor: a position (e.g. start position and length of span)Computation: the text of the span to search for
- Object drawings:
  - Object ID: built in graphical object ID
- Bitmaps
  - (Object ID: ID of graphical object in transparent layer)
  - Structure Descriptor: Coordinates of seleciton shapes
- Video and Sound
  - (Object ID: ID of graphical object in transparent layer)
  - Structure Descriptor: segment (e.g. start time and length of segment)

Handling link inconsistency - example

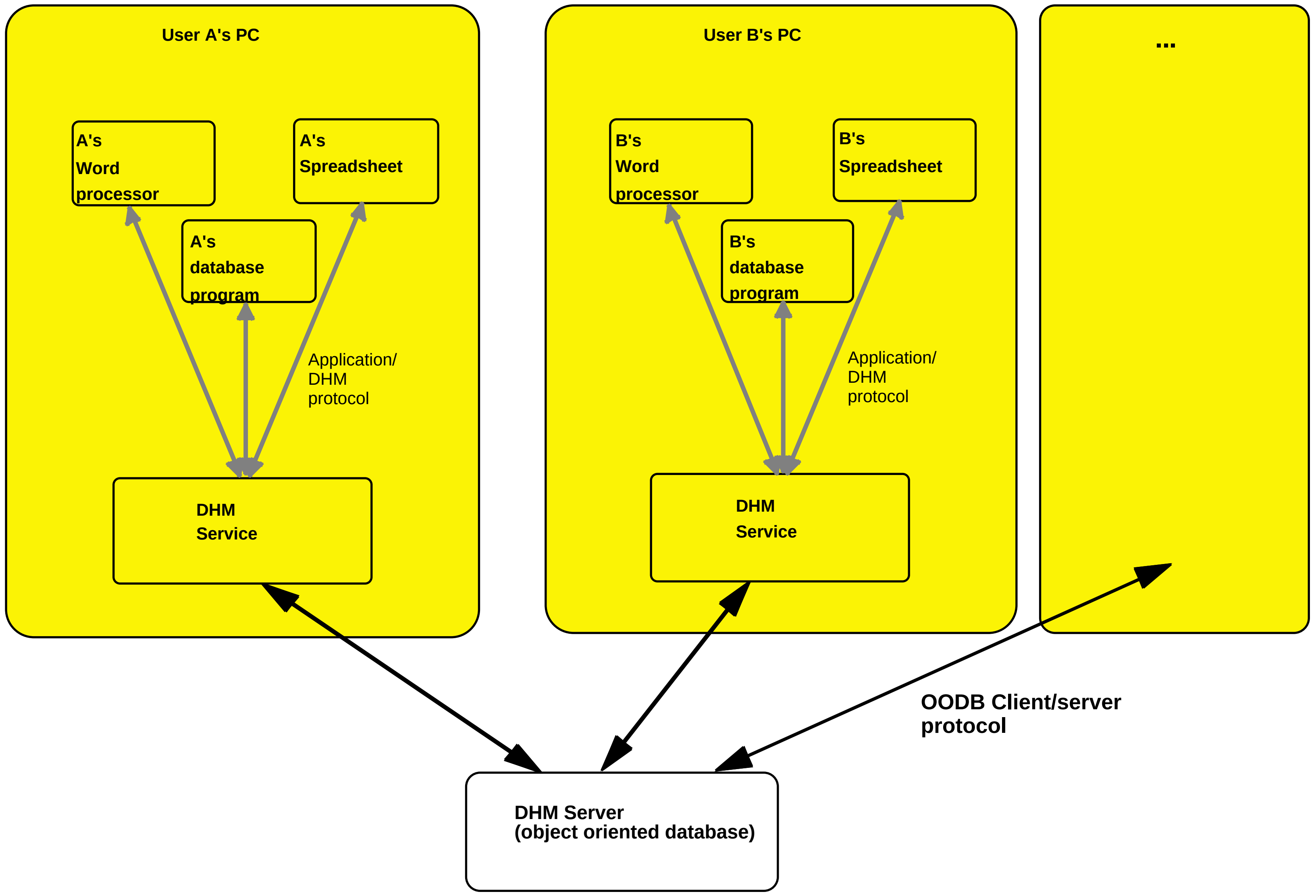
A text editor supporting bookmarks (Word, WordPerfect etc.)

1. Tell the user that the bookmark with the given ID does not exist anymore in the document
2. Get the word processor to highlight the span of text corresponding to the Structure descriptor and tell the user that the old bookmark represented this span.
3. Tell the user that the span corresponding to the structure descriptor is not consistent with the stored text string in the Computation attribute.
4. Ask the user whether she wish to update the anchor to locate this new span or she wish to perform a search for the the text string of the Computation attribute.
5. Let the user perform a search for the text string of the Computation attribute.
6. If the text string is found: Ask the user whether she wish to update the anchor to locate the span of text in this new posistion.

Changing access rights

1. When a document is read-only, only fill in the Structure Descriptor and the Computation criteria of Anchor LocSpecs
2. When a document is writeable, create persistent selections and fill in all attributes of the Anchor LocSpecs, including the Object ID for the persistent selections.
3. When a read-only document becomes writeable, then the Runtime layer sends a message with all the LocSpecs to create persistent selections for all the “read-only” LocSpecs, before any edits take place.
4. When a writeable document becomes read-only, then the Runtime layer switches to generation of “read-only” LocSpecs with empty Object ID attribute.

Devise Hypermedia: Multi-user architecture



Devise Hypermedia: experiences from several platforms

- ■ Windows/NT (forthcoming Windows 95)
  - Microsoft Word(6.0) and Excel(5.0), Intergraph’s Microstation (CAD)
  - In progress: WordPerfect
- ■ Apple Macintosh (68K and PowerPC)
  - Microsoft Excel (4.0)
  - In progress: Microsoft Word(6.0) and Excel(5.0)
- ■ Sun Solaris (Unix)
  - Mjølner Sif (old implementation statically linked)
  - vi, Design/CPN, ORACLE database views, Rank Xerox’s Ariel/ Documentum,
  - In progress: Emacs

Wrap up

- We have taken significant steps towards providing an extensible Dexter-based hypermedia service to third party applications
  - basic anchor based linking can be provided
  - experiences from several platforms and application types

But there is still open issues

- How to make the linking service more application independent?
  - standardized use of anchor and PSpec values
- How to combine the multi-user facilities of the hypermedia service with filesystem access control and sharing?
- How to support cross platform hypermedia support for external files and third-party appplications?
-