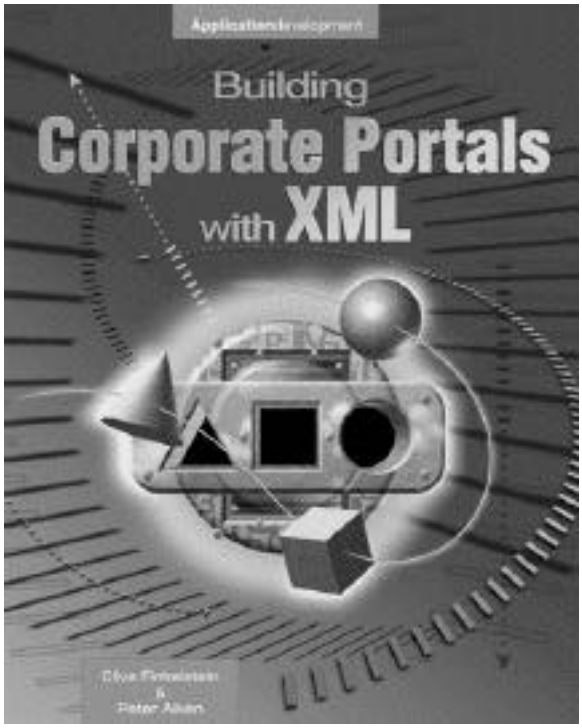


Building Corporate Portals Using XML

Peter Aiken

Director Institute for Data Research
Associate Professor of Information Systems
Virginia Commonwealth University
Richmond, VA 23284-4000

paiken@acm.org
804.828.0174
http://fast.to/peteraiken

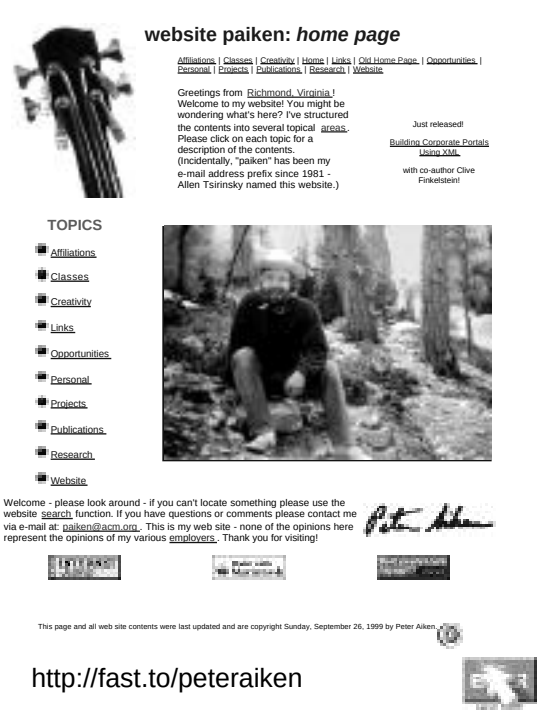
Peter Aiken

Research Areas:
reengineering, data reverse engineering, software requirements engineering, information engineering, human-computer interaction, systems integration/systems engineering, strategic planning, and decision support systems

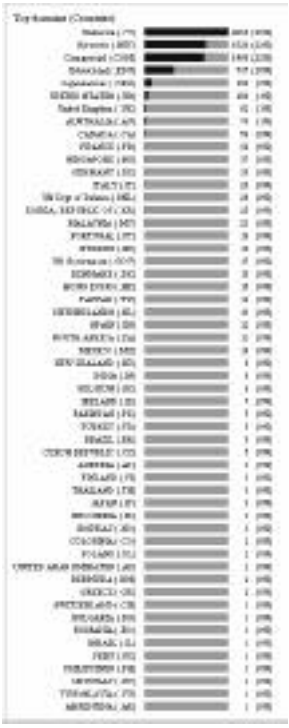
- Full time in information technology since 1981
- Information technology engineering research and project background
- University-level teaching experience since 1979
- Director: George Mason University/Hypermedia Technologies Laboratory (1989-1993)
- DoD Computer Scientist: Reverse Engineering Program Manager/Office of the Chief Information Officer (1992-1997)
- Published Papers in: C***ommunications of the ACM***, ***IBM Systems Journa***l, ***InformationWEEK***, ***Information & Management, Information Resources Management Journal, Hypermedia***, ***Information Systems Management***, ***Journal of Computer Information Systems*** and ***IEEE Software***
- Associate Professor/Information Systems & Director/Institute for Data Research

Consulting/Research Experiences

- Apple Computer
- US Department of Defense
  - Army Corps of Engineers
  - Strategic Defense Initiative
  - Defense Information Systems Agency
  - Corporate Information Management
  - Joint Logistics Commanders/Joint Group on Systems Engineering
  - Office of the Secretary of Defense
- The Analytical Science Corporation
- University of Guam
- Edison Technology Center/Cuyahoga Community College
- Virginia Center for Innovative Technology
- Information Engineering Systems Corporation
- Commonwealth of Virginia - Departments of
  - Personnel and Training
  - Transportation
  - Social Services
  - Business Assistance
  - Corrections
- Time Life, Incorporated
- Innovative Business Solutions, Inc.
- Circuit City Stores
- Deutsche Bank/Global Settlement Technologies Group
- Mattel Toys
- NationsBank/Bank of America
- Evoke Software
- Knowledge Partners Inc.

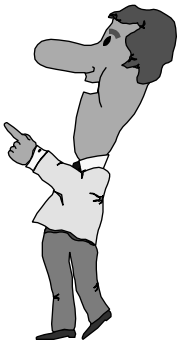


http://fast.to/peteraiken



Overview

"Focused use of advanced technologies for knowledge management"



- Legacy Systems
- Portals Definition
- Development Example
- Metadata Engineering
- XML as a Portal Component
- Advanced Portal Development Technologies
- An ERP-based Example

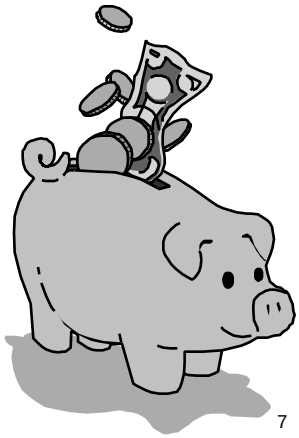
Drawn from:

- Aiken, P. H., Ngwenyama, O. K., and Broome, L. Reverse Engineering New Systems. IEEE Software. May-June, 1999.
- Aiken, P.H., Yoon, Y., and Leong-Hong, B., Requirements Driven Data Engineering. Information & Management February 1999 35(3):155-168.
- Aiken, P.H., Reverse engineering of data. IBM Systems Journal, 1998. 37(2): p. 246-269.

- Aiken, P.H., A. Muntz, and R. Richards, DoD Legacy Systems: Reverse Engineering Data Requirements. Communications of the ACM, 1994. 37(5): p. 26-41.
- Aiken, P.H., Data Reverse Engineering: Slaying the Legacy Dragon. 1996, New York: McGraw-Hill. 394 pages.
- Finkelstein, C. and Aiken, P. Enterprise Information Portals - Data Warehousing And Decision Support: Knowledge Management and the Internet. 1999 McGraw-Hill, 400 pages.

A Legacy Example

- **Cash Management System (CMS) Functional Summary**
- Supports check processing and other specialized services for large corporate customers:
  - Zero balance account
  - Reconciliation of cleared checks
  - Electronic funds transfer (Swift)
  - Lock box operations
  - On-line query facility
  - Cost: $16 M/annually



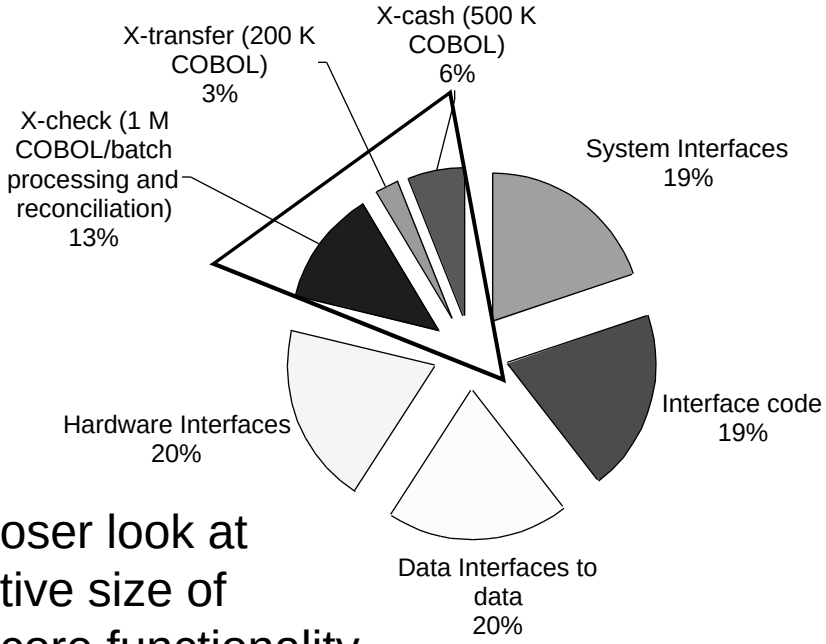
Example from: Michael L. Brodie & Michael Stonebraker ***Migrating Legacy Systems: Gateways, Interfaces & The Incremental Approach*** Morgan Kaufmann Publishers, 1995

CMS Technical Overview

- Most built in 1981 - 100 gigbytes
- 40 software modules
- 8 million lines of code
- COBOL/CICS/VSAM on IBM 3090
- Federal Reserve Bank connection Tandem (TAL)
- Lock box uses VAX (C)
- Processes 300,000 transactions daily & 1-2 million checks nightly
- Generations of programmers

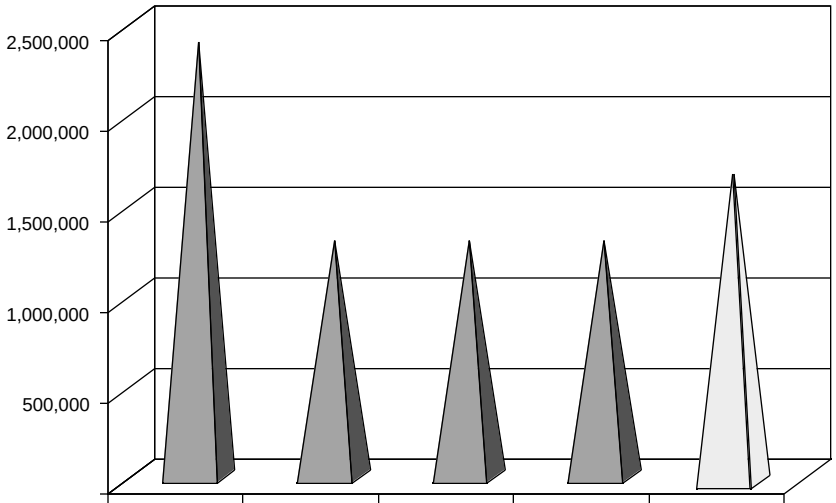
Peeling

- 80 - 20 rule
- Slice away 6.3 of 8 LOCs as non-key functionality
- Xcheck (1 M COBOL/batch processing and reconciliation)
- Xtransfer (200 K COBOL)
- Xcash (500 K COBOL)
- Remainder
  - interfaces to other organizational parts
  - interface code
  - Interfaces to data
  - Interfaces to hardware (Check sorting hardware)



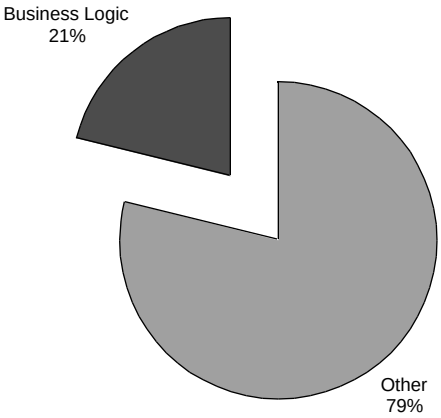
A closer look at relative size of the core functionality

Another perspective (what's bigger than the combined "important" components?)



Interface code	System Interfaces	Data Interfaces	Hardware interfaces	Three important components:
2,400,000	1,300,000	1,300,000	1,300,000	1,700,000

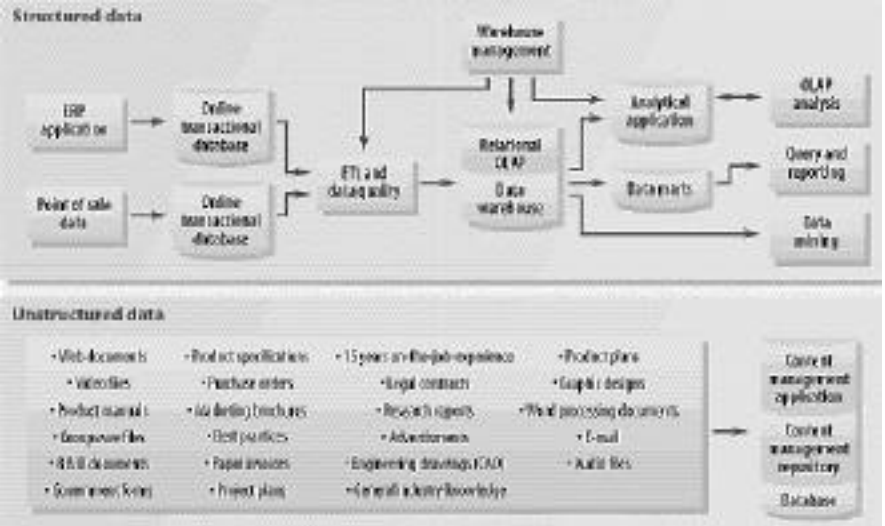
Business logic is contained in just 21% of the code



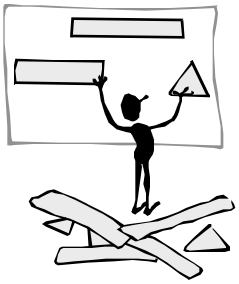
Portals Defined

- A ***style*** of developing information delivery systems
- Three key elements:
  - Engineered, XML-based and metadata-based data integration
  - Internet, Intranet, TCP/IP-based interfaces and delivery
  - Extensive use of new technologies including
    - » 4GLs
    - » Data analysis tools
    - » Business rule engines
    - » Data logistic networks
- ***Users won't know or really care about any of the above!***

Possible EIP Components



Portal Planning Characteristics



No step > 10 person-years

No step > 1 calendar year

Each step <= 1 year payback

Hypothetical Development Example

World-Wide Airlines Online
Frequent-Flyer Statement

World-Wide Airlines Online Frequent-Flyer Statement

- 500,000 frequent-flyer statements mailed monthly
- Integrates information from
  - Charge card purchases
  - Long distance charges
  - Rental car charges
  - Hotel charges
- Strategic advantage over the competition
- Program participants tend to be detail-oriented
- Demanding web access to statements

World-Wide Airlines Frequent Flyers Mileage Summary

Identification Number:	123 456 789
Additional miles to earn a free ticket:	4,099
Mileage Credited: 02SEP99 to 30SEP99:	7,471
1999 mileage accumulation:	(*) 35,901
Mileage since enrollment:	215,901

(*) Includes unredeemed mileage carried over from 1998

ACTIVITY SUMMARY:

Date	Flight	Description of Mileage	Credit	Miles	Bonus
13SEP99		Long-distance Mileage	60		
14SEP99		Bank-card Mileage	1393		
24SEP99	288	San Diego, CA-Detroit, MI		1956	
24SEP99		30% Exclusive, Bonus			587
24SEP99	474	Detroit MI-Richmond, VA		750	
24SEP99		30% Exclusive, Bonus			225
25SEP99		Rent A Car-San Diego	500		
26SEP99		San Diego Hotel	2000		

World-Wide Airlines Online Frequent-Flyer Statement

Proposed Steps

- Reverse engineer MARS
- Obtain metadata information
- Analyse metadata
- Form the basis for the reengineering project

World-Wide Airlines Online Frequent-Flyer Statement

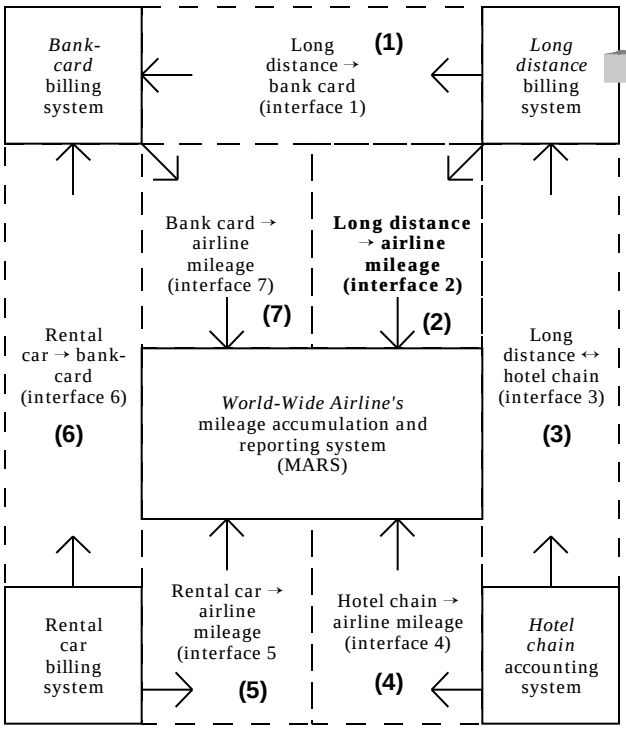
Project scope:

- Define common information exchange formats and technologies
- Initially, the project will require participation from the long-distance company
- Eventually, all our business partners will be drawn into this effort

World-Wide Airlines Online Frequent-Flyer Statement

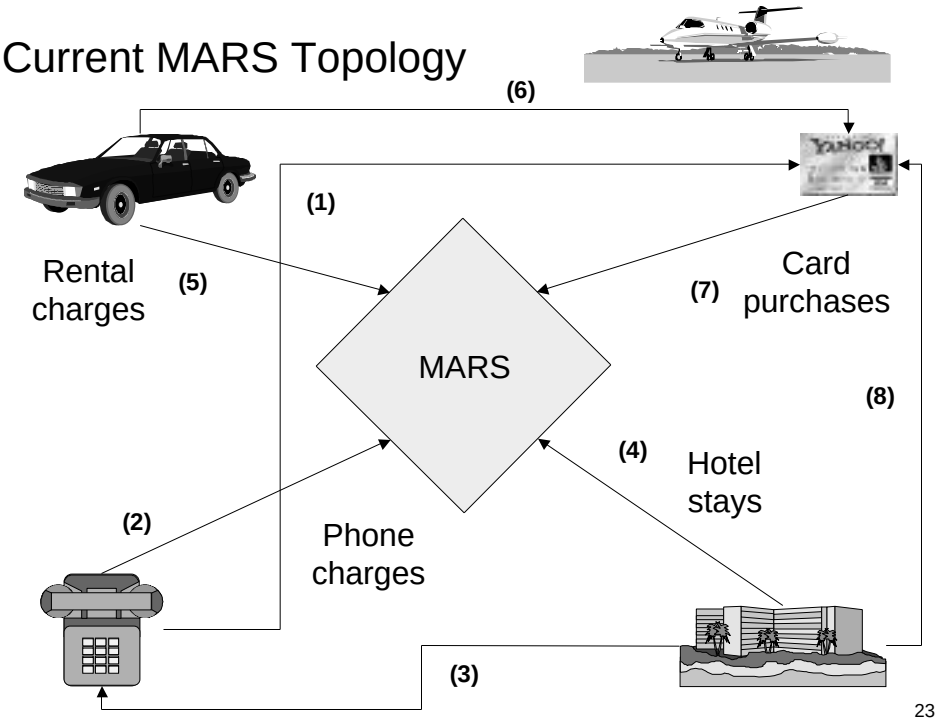
Anticipated results: within 3 months

- Reverse Engineering and Analyze
  - System metadata
  - Business rules
  - System requirements
- Deliver
  - Duplicate internet statement
  - Historical records dating back previous 12 month
  - Search capabilities

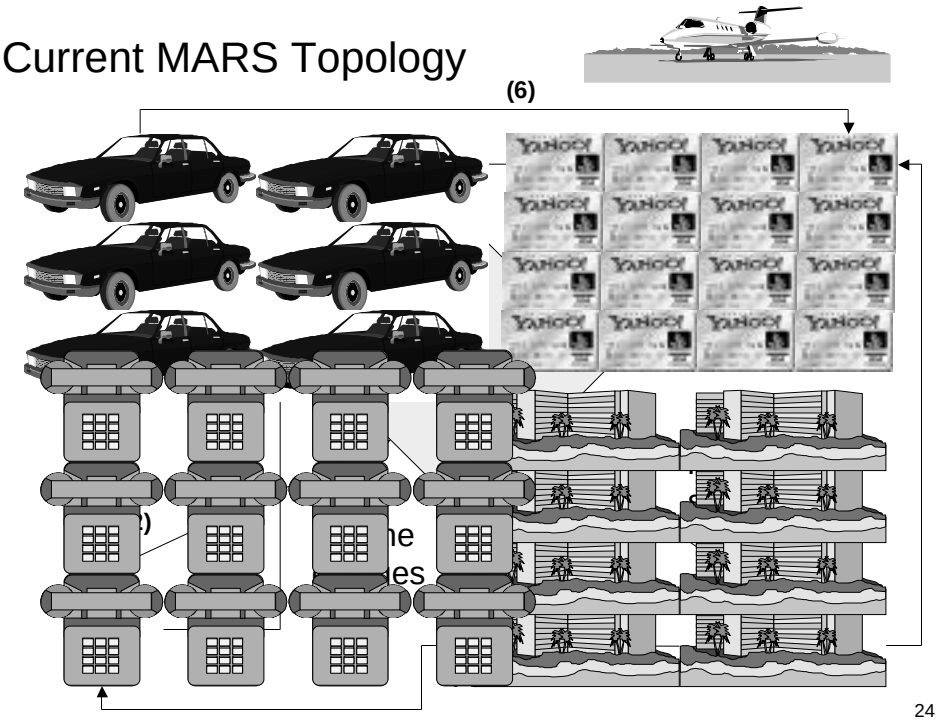


Interfaces required to support information exchange among five different organizations to produce frequent-flyer statements

Current MARS Topology



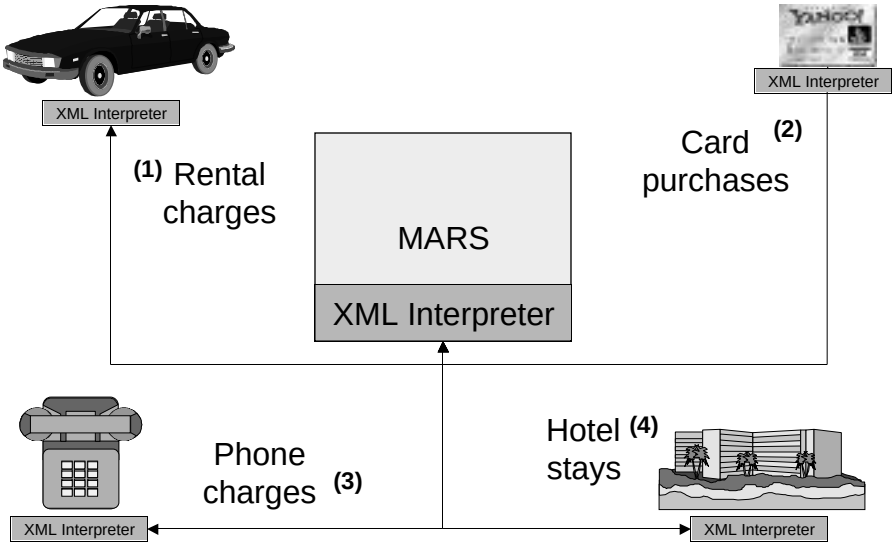
Current MARS Topology



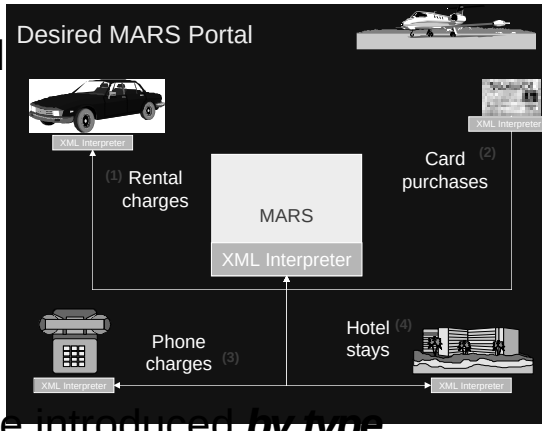
MARS Functionality

Integrate different data formats for each partner company
Apply different reward rates (within and across partners)
Distinguish different status program participants
Integrate rotating promotional offers
Integrate with world wide airline booking systems
Integrate with affiliated world wide airline web site
Print accurate and timely statement packages
Be maintained on different platforms around the world

Desired MARS Portal



Desired MARS Portal

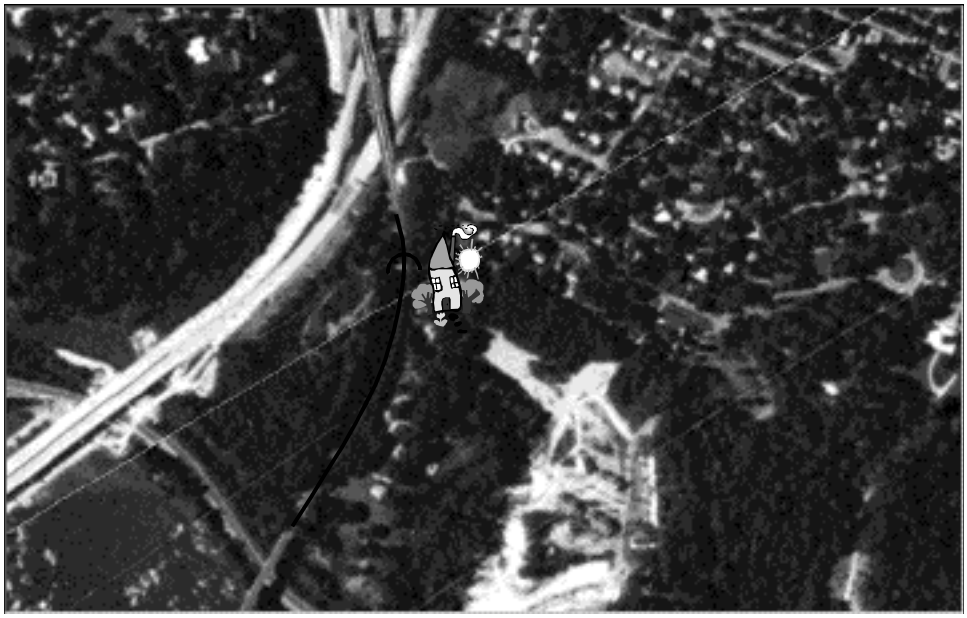


- Reduced interfaces from ***7 types*** to four
- Each interface is bi-directional
- New interfaces are introduced ***by type***
- Data is XML-wrapped and applications are XML-enabled
- Changes to data and/or structures are encoded, not hard coded

Engineering

- fundamentally concerned with the design and manufacture of complex products using calculated manipulation or direction
- originally divided into:
  - military & civil engineering
- over time:
  - steam engineering
  - gas engineering
  - agricultural engineering
  - topographical engineering
  - electrical engineering
- now includes:
  - chemical, electronic, aerospace, urban, telecommunication, software and systems

My Back Yard (and front yard also)



If software engineers were responsible for that type of engineering

Video clip from ***The Fugitive*** (1993)

Bonfire Collapse Conclusions

- Too much stress on logs
- Linbeck said two primary factors caused the failure:
  1. Excessive internal stresses caused by aggressive wedging of the second stack of logs into the first stack.
  2. Inadequate containment strength. The wiring used to tie the logs together provided insufficient binding strength. Also, steel cables that in recent years had been wrapped around the first stack of logs were not used in 1999.
- Linbeck said the commission ruled out several factors that had been considered as possible contributing factors for the collapse:
- The bonfire's center pole. Linbeck said the "center pole, which was of high quality, had very little to do with the structural strength" of the stack. The center pole "did not contribute to the collapse."
- A crane that hit a cross-tie. The committee verified that the crane hit the stack of logs, but Linbeck said it "could not have generated enough force to materially weaken the structure or contribute to the collapse."
- Soil under the logs, ropes used to secure the logs and other equipment. Linbeck said all were tested. "None of them played a role in the collapse," he said.

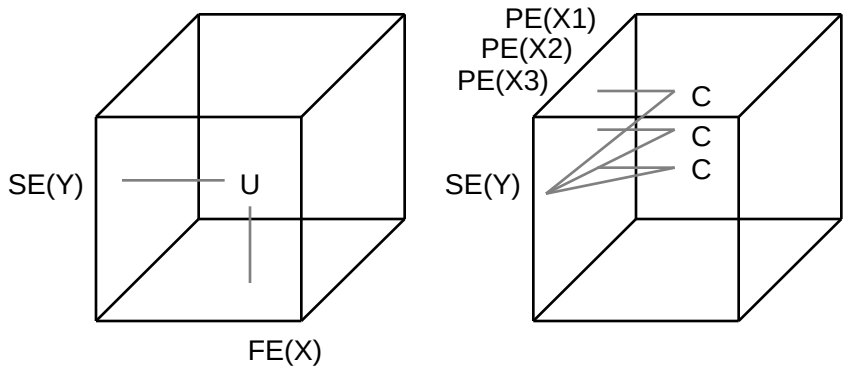
http://www.tamu.edu/bonfire-commission/reports/Packer_Report.pdf

http://www.tamu.edu/bonfire-commission/reports/Packer_Report.pdf

Bonfire Collapse

Multidimensional Data Source

- Metadata describing system data can considered as a multidimensional data



Screen Element (Y)
updates
File Element (X)

Screen Element (Y)
creates
Printout Elements (X1,X2,X3)

Metadata is used to maintain information about system data

Printout Element		Use		Screen Elements
PE-43				
PE-44		create		
PE-45		read		
PE-46	is used for	update	by	SE-255
PE-47		delete		SE-434
PE-48				SE-756
PE-49				SE-1056
PE-50				

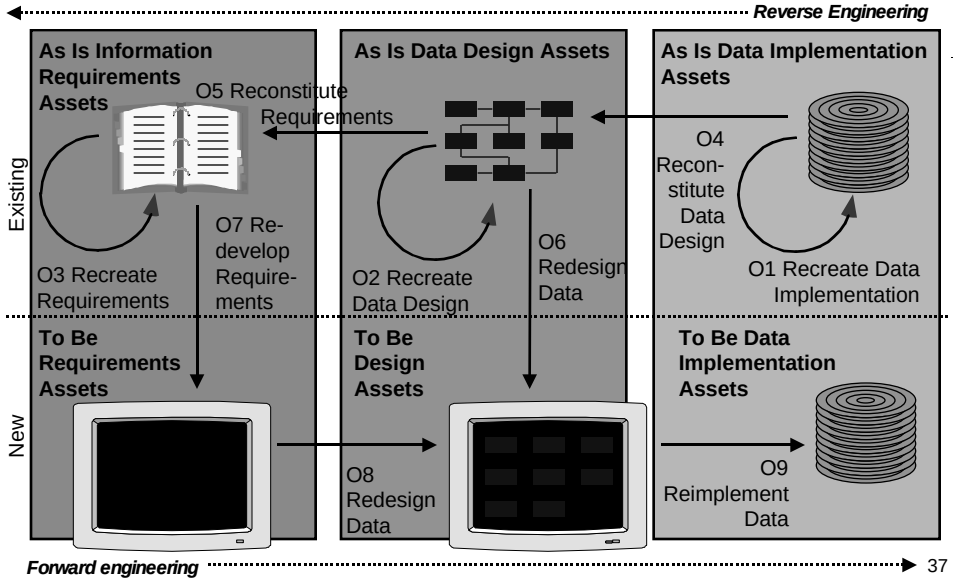
How information provided by metadata engineering enhances data reengineering

- Uses structured techniques to reconstitute the data assets of an existing system in situations where the system metadata has deteriorated or become confused
- Accomplished by identifying interrelated system data components and creating representations of them in another form or at a higher level of abstraction
- Typically models of the data implementation, data design, or system data requirements
- Metadata engineering and data reengineering are inextricably linked

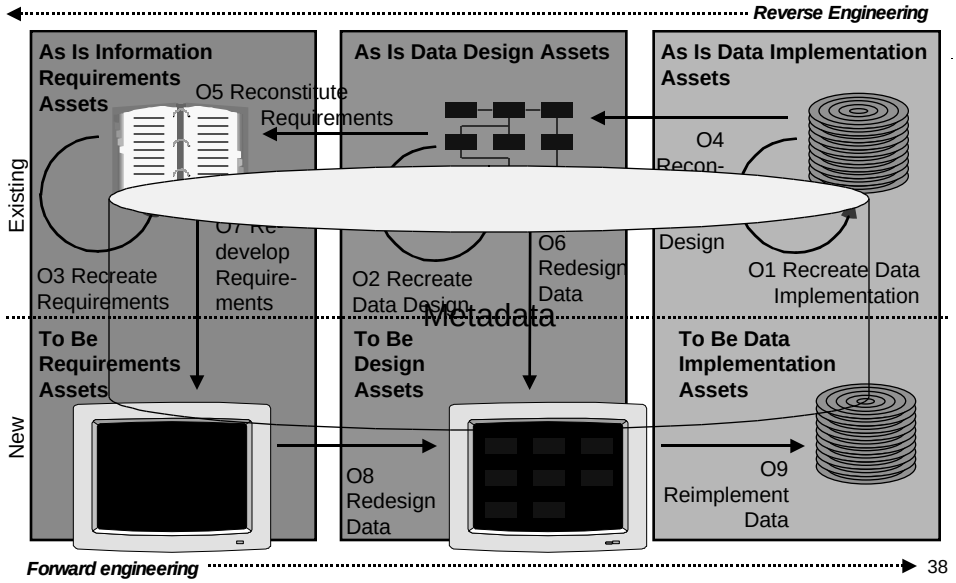
Metadata and Reengineering

- Structured means of informing the transition from existing (or as is) data assets to new, more useful (to be) data assets
- Based on coordinated reverse and forward engineering activities
  - where forward engineering benefits from metadata gained from the reverse engineering
- metadata engineering enables data reengineering
- Three classes of data reengineering options:
  - Reconstitute original metadata
  - Improve the current metadata
  - Improve system data capabilities based on the improved metadata
  - Three classes of options are comprised of nine individual options

Metadata Engineering



Metadata Engineering



Metadata Engineering Options

O-1 data implementation (e.g., by recreating descriptions of implemented file layouts);

O-2 data designs (e.g., by recreating the logical system design layouts); or

O-3 information requirements (e.g., by recreating existing system specifications and business rules).

O-4 data design assets by examining the existing data implementation (when appropriate O-1 can facilitate O-4); and

O-5 system information requirements by reverse engineering the data design O-4. (Note: if the data design doesn't exist O-4 much precede O-5.)

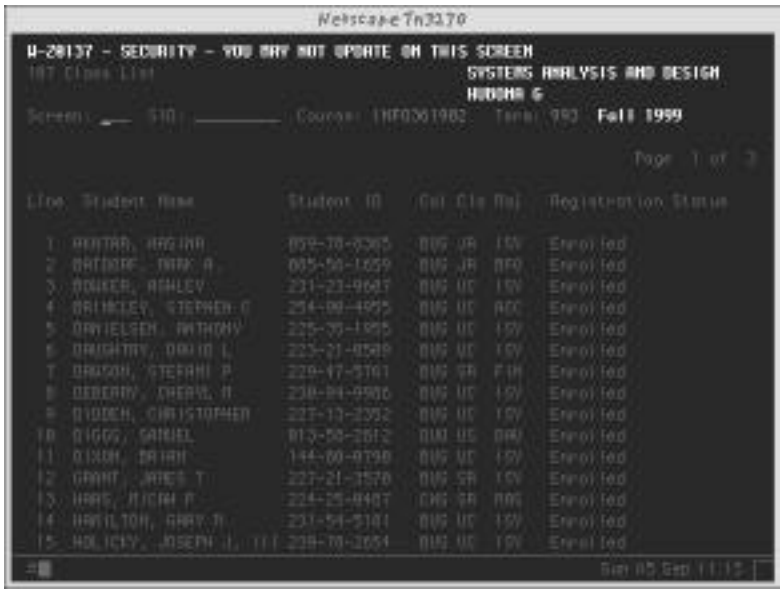
O-6 transforming as is data design assets, yielding improved to be data designs that are based on reconstituted data design assets produced by O-2 or O-4 and (possibly O-1);

O-7 transforming as is system requirements into to be system requirements that are based on reconstituted system requirements produced by O-3 or O-5 and (possibly O-2);

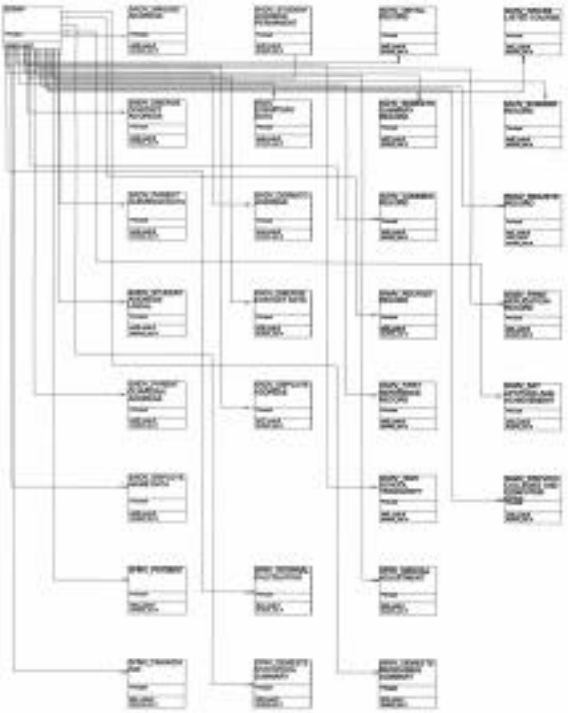
O-8 redesigning to be data design assets using the to be system requirements based on reconstituted system requirements produced by O-7; and

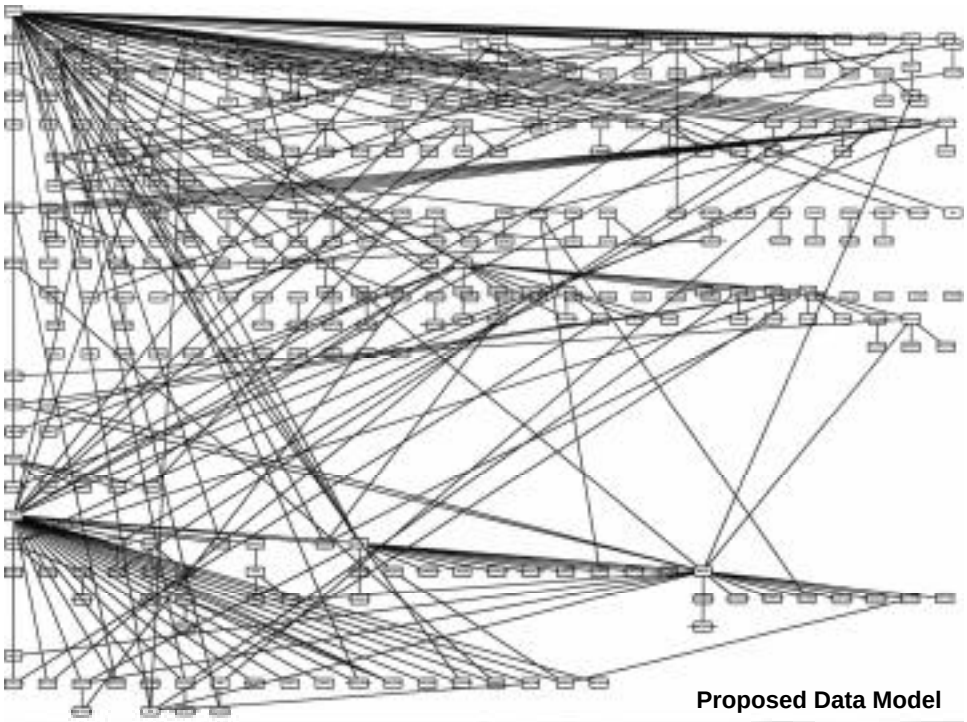
O-9 re-implementing system data based on data redesigns produced by O-6 or O-8.

Why data structure problems have been difficult?



Student System Data Model



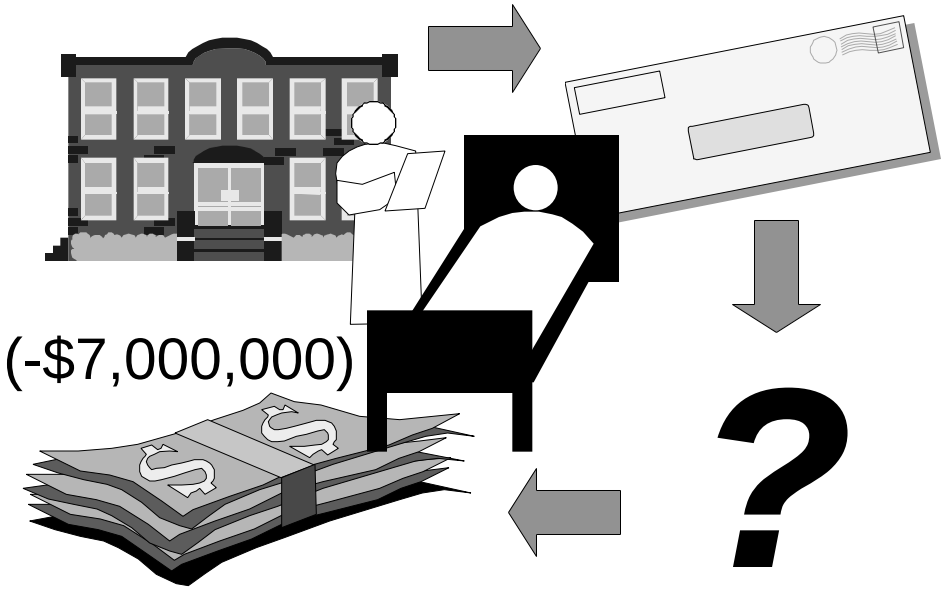


Proposed Data Model

Other Organizational Assets

Cash and other financial instruments
Real property
Inventory
Intellectual Property
Human knowledge, skills, and abilities
Organizational reputation and good will

One way to value data



Java, XML = Similar Excitement

Why the excitement over Java?

- Write it once
- Play it back anywhere

Why the excitement over XML?

- Wrap it once
- Utilize it anywhere
- XML is electronic data interchange (EDI) for the rest of us
- XML is to data what Java is to programming languages

Anyone Remember Wordstar?

- To change text attributes you embedded tags into the text. For example if you wanted the next word to be bolded, you would add the tag before the next word. To turn the bolding off, you used a to remove future bolding. Similarly, <u> began underlining and </u> ended underlining formatting. Italics was <i>, etc.

Interpreted Text

- To change text attributes you embedded tags into the text. For example if you wanted the next word to be bolded, you would add the tag **before the next word. To turn the bolding off, you used a** to remove future bolding. Similarly, began underlining and ended underlining formatting. Italics was***, etc.***

HTML vs. XML

```
<html>
  <head>
      <title>Test Web Page</title>
  </head>

  <body>
  This is the body of the web page. <br>
  I've also added a picture of me playing bass. <br>
  <img src="playingbass.gif">
  </body>

</html>
```


HTML vs. XML

HTML Drawbacks

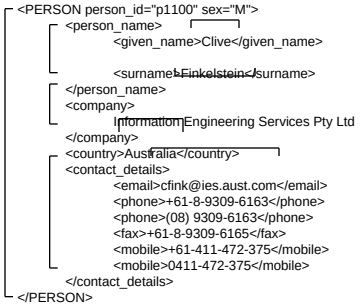
1. **No effective way to identify content of page:** HTML tags describe the layout of the page. Web browsers use the tags for presentation purposes, but the actual text content has no specific meaning associated with it. To a browser, text is only a series of words to be presented on a Web page for display purposes.
2. **Problems locating content with search engines:** Because of a lack of meaning associated with the text in a Web page, there is no automatic way that search engines can determine meaning – except by indexing relevant words, or by relying on manual definition of keywords.
3. **Problems accessing databases:** We discussed earlier that Web pages are static. But when a Web form provides access to online databases, that data needs to be displayed dynamically on the Web page. Called "Dynamic HTML" (DHTML), this capability enables dynamic content from a database to be incorporated ``on the fly'' into an appropriate area on the Web page.
4. **Complexity of dynamic programming:** DHTML requires complex programming to incorporate dynamic content into a Web page. This may be written as CGI, Perl, ActiveX, JavaScript, or Java logic, executed in the client, the Web server, the database server, or all three.
5. **Problems interfacing with back-end systems:** This is a common problem that has been with us since the beginning of the Information Age. Systems written in one programming language for a specific hardware platform, operating system, and DBMS may not be able to be migrated to a different environment without significant change or a complete rewrite. Even though it is an open architecture, HTML also is affected by our inability to move these legacy systems to new environments.

HTML vs. **XML**

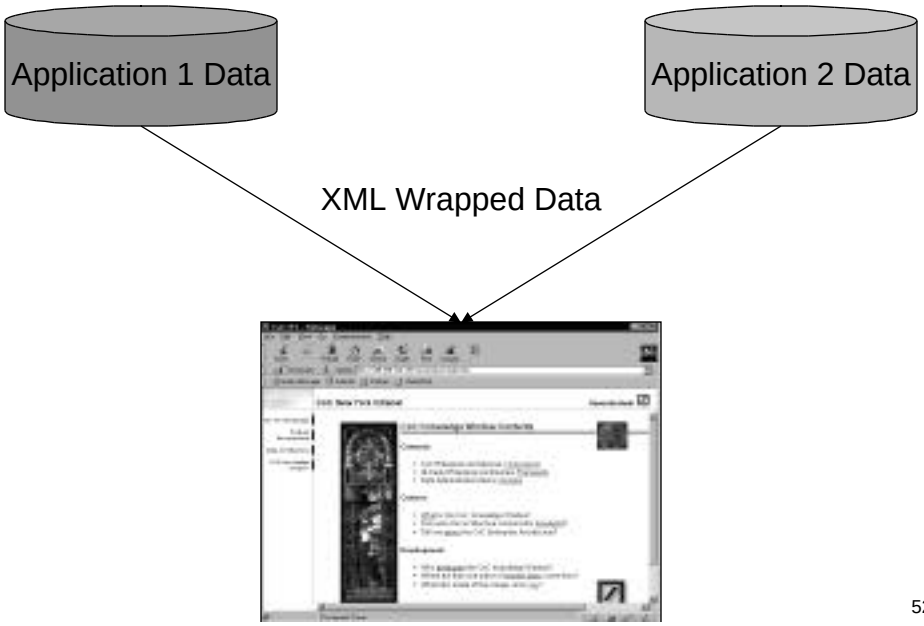
- Formats can be specified for any application
- XML compliant browsers can correctly interpret any data sent to it
- XML compliant applications can be used to interpret any type of data

Figure 1.1: An example of an XML document with metadata tags (surrounded by < … >) identifying the meaning of following data

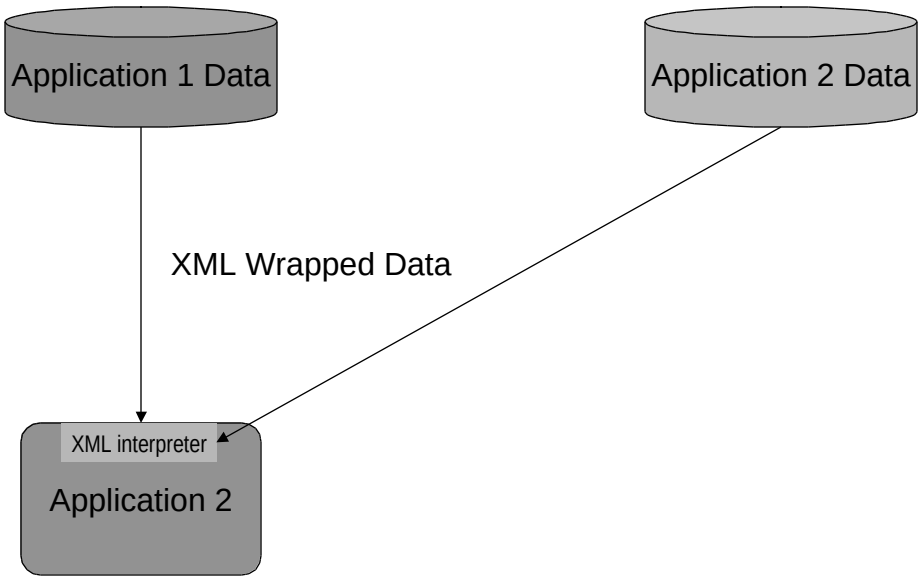
```
<PERSON person_id="p1100" sex="M">
        <person_name>
                <given_name>Clive</given_name>
                <surname>Finkelstein</surname>
        </person_name>
        <company>
                Information Engineering Services Pty Ltd
        </company>
        <country>Australia</country>
        <contact_details>
                <email>cfink@ies.aust.com</email>
                <phone>+61-8-9309-6163</phone>
                <phone>(08) 9309-6163</phone>
                <fax>+61-8-9309-6165</fax>
                <mobile>+61-411-472-375</mobile>
                <mobile>0411-472-375</mobile>
        </contact_details>
</PERSON>
```



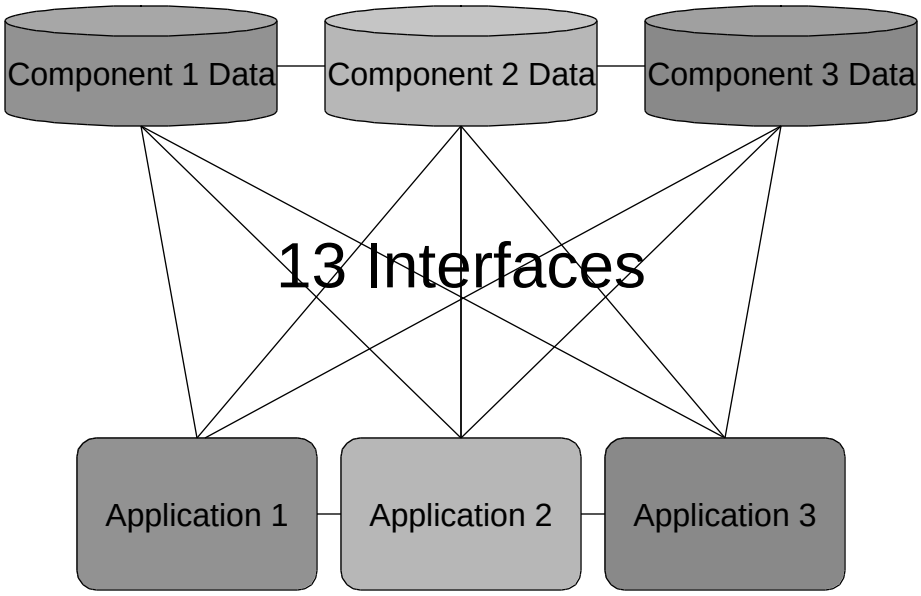
XML Integration at the Browser



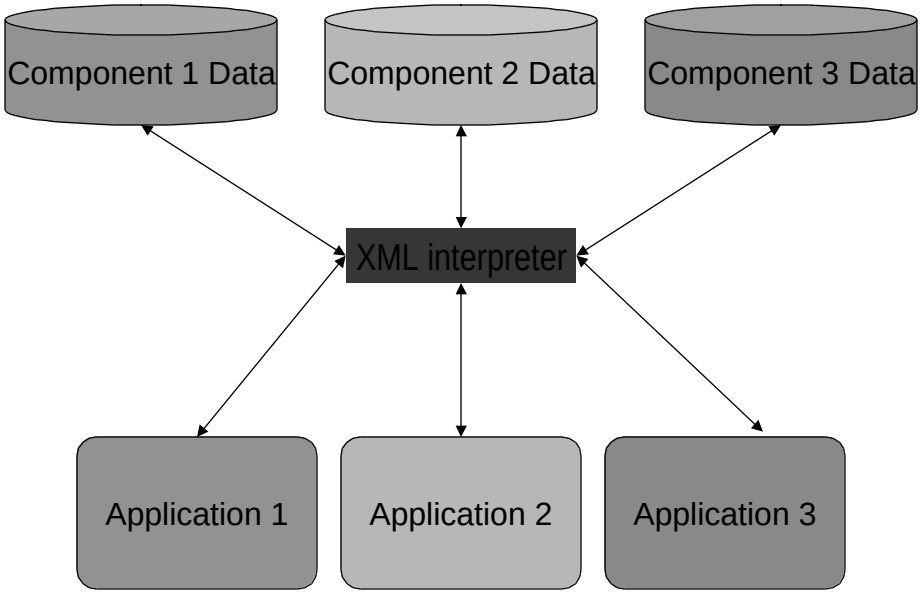
XML-based Application Integration



What is required to solve this component integration problem?



XML-based Component Integration



XML Components

XML **Extensible Markup Language** Defines document content using metadata tags and namespaces

DTD **Document Type Definition** Defines XML document structure (analogous to DDL schema)

XSL **Extensible Style Language XSL or Cascading Style Sheets** (CSS) separate layout from data

XLL **Extensible Linking Language** XLL implements bi-directional links (single or multiple)

DOM **Document Object Model** Implements a standard API for processing XML in any language

RDF **Resource Description Framework** W3 Interoperability Project for data content interchange

XML Example

```
[createwholefile]
<ALLQUOTES>
    <QUOTE>
        <COMPANY ID=17>House of Hardware</COMPANY>
        <DATE>May 6, 9:00</DATE>
        <PRODUCT>M8 metricwingnut, steel, zinc</PRODUCT>
        <PRICE>$7</PRICE>
        <AVAILABLE>2000</AVAILABLE>
    </QUOTE>
    <QUOTE>
        <COMPANY ID=23>Nutz 'N Such</COMPANY>
        <DATE>May 6, 9:02</DATE>
        <PRODUCT>M8 metricwingnut, steel, zinc</PRODUCT>
        <PRICE>$6</PRICE>
        <AVAILABLE>1500</AVAILABLE>
    </QUOTE>
    <QUOTE>
        <COMPANY ID=07>Hardware Haven</COMPANY>
        <DATE>May 6, 9:04</DATE>
        <PRODUCT>M8 metricwingnut, steel, zinc</PRODUCT>
        <PRICE>$4.50</PRICE>
        <AVAILABLE>500</AVAILABLE>
    </QUOTE>
```


XML Example

Company ID	Price	Availability
House of Hardware	$7.00	2000
Pleasants Hardware	$6.25	250
Howards Supplies	$6.59	1250
Nutz 'N Such	$6.00	1500
Hardware Haven	$4.50	500
Howard's Supplies	$8.33	200
Loew's	$7.50	150
Ed's Builders	$6.75	1000

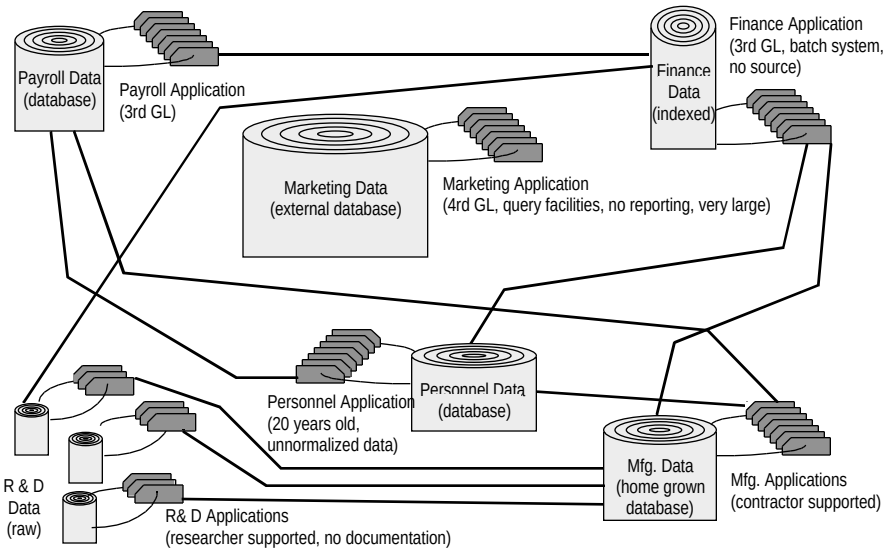
XML Example

Company ID	***Price***	Availability
Hardware Haven	$4.50	500
Nutz 'N Such	$6.00	1500
Pleasants Hardware	$6.25	250
Howards Supplies	$6.59	1250
Ed's Builders	$6.75	1000
House of Hardware	$7.00	2000
Loew's	$7.50	150
Howard's Supplies	$8.33	200

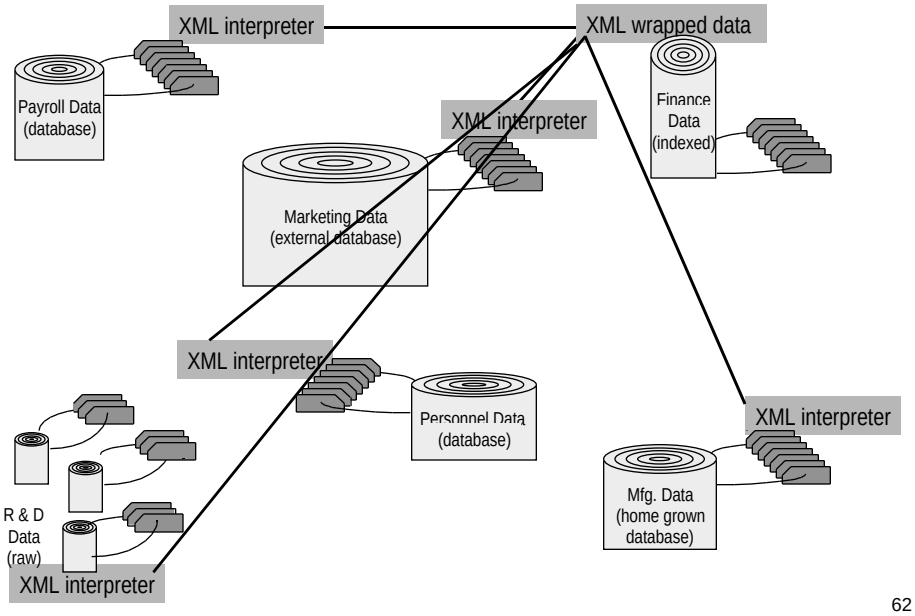
XML Example

Company ID	Price	***Availability***
Loew's	$7.50	150
Pleasants Hardware	$6.25	250
Howard's Supplies	$8.33	200
Hardware Haven	$4.50	500
Ed's Builders	$6.75	1000
Howards Supplies	$6.59	1250
Nutz 'N Such	$6.00	1500
House of Hardware	$7.00	2000

Existing Situation



Becomes this ...



XML References

- An initial introduction to XML (and also Cascading Style Sheets) is provided by “XML: A Primer” [St Laurent 1998].
- XML used for web site development, with HTML, XSL and XLL, is addressed in “XML: Extensible Markup Language” [Harold 1998].
- “XML Complete” [Holzner 1998] covers the use of XML with Java.
- “Web Farming for the Data Warehouse” [Hackathorn 1998] uses the Internet, Intranets and XML for access to external data sources for warehouse deployment.

XML Information Web Sites

WWW Consortium - Specifications and Standards
- http://www.w3.org/

Microsoft XML Scenarios Web Site
- http://microsoft.com/xml/scenario/intro.asp

XML.com Web Site
- http://www.xml.com/

James Tauber’s XMLINFO Web Site:
- http://www.xmlinfo.com/

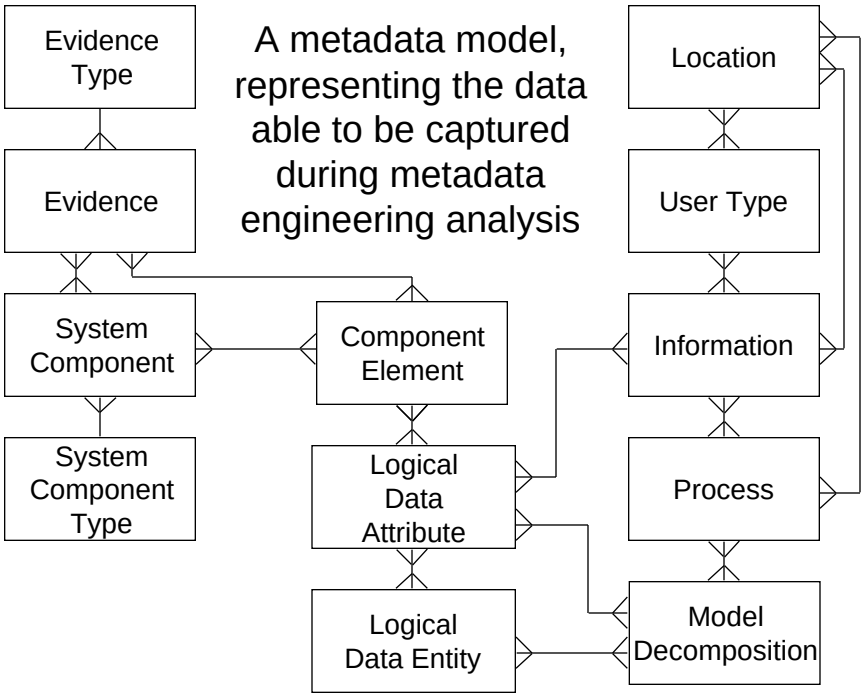
James Clark’s XML Web Page
- http://www.jclark.com/xml/

Robin Cover’s XML Resources
- http://www.sil.org/sgml/xml.html

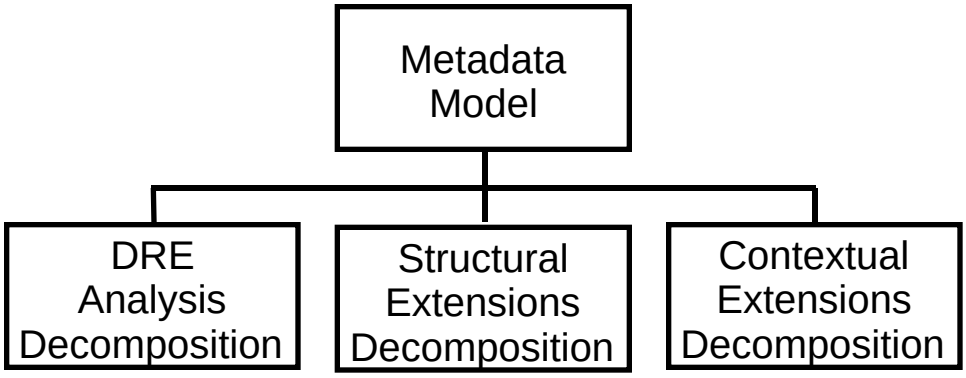
Microsoft XML Site
- http://www.microsoft.com/xml/

Microsoft XML Workshop Web Site
- http://www.microsoft.com/workshop/xml/toc.htm

Web Farming Web Site
- http://webfarming.com/



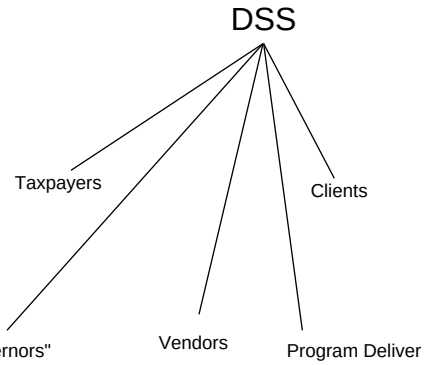
Three Metadata Model Views



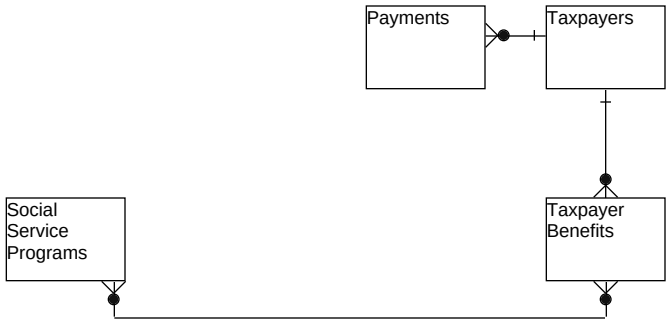
Data model is comprised of model views

DSS Strategic Data Model

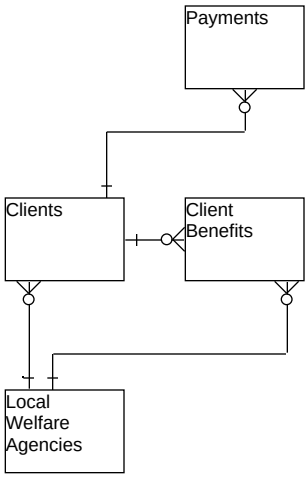
- Taxpayer view
- Client view
- Governance view
- Program Delivery view
- Vendor view



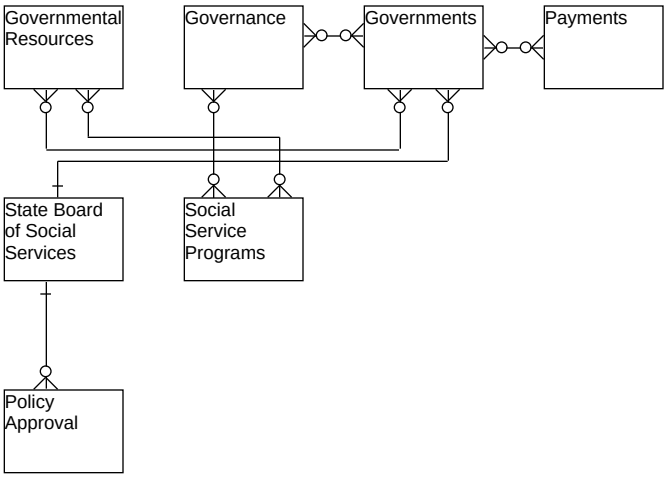
Taxpayer view



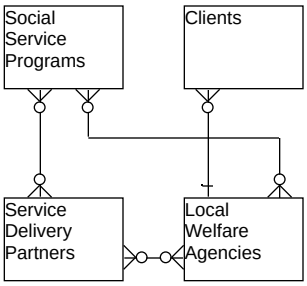
Client view



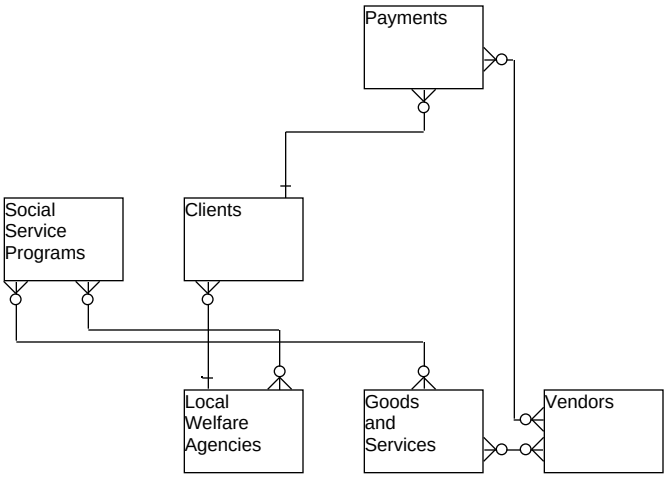
Governance view



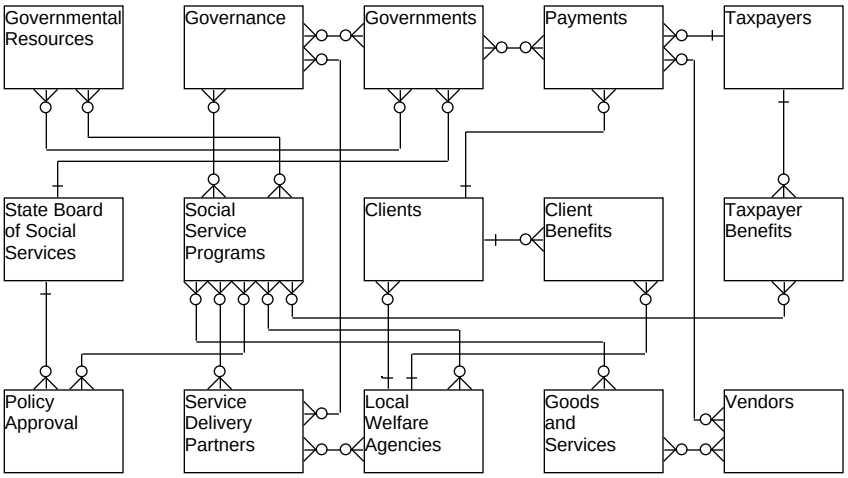
Program Delivery view



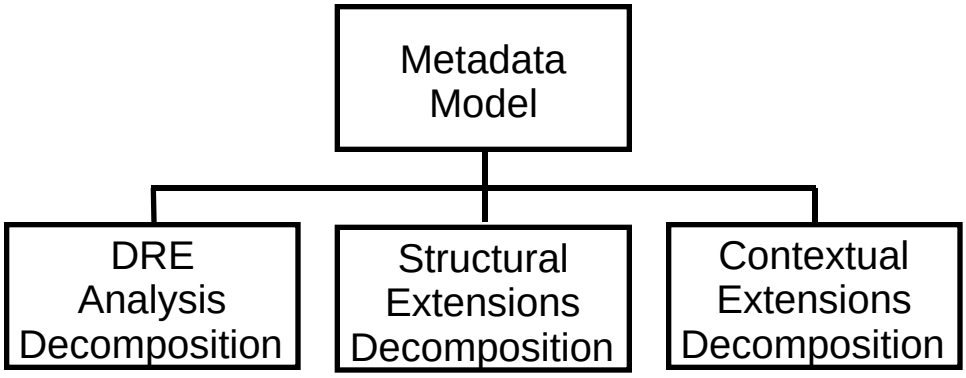
Vendor view

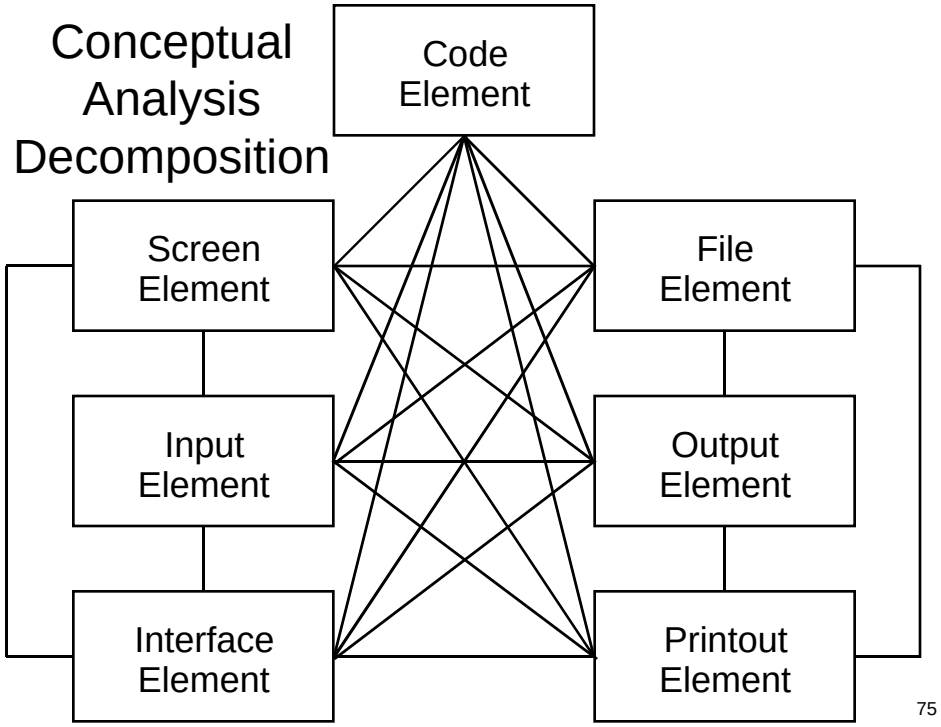


DSS Strategic Level Data Model

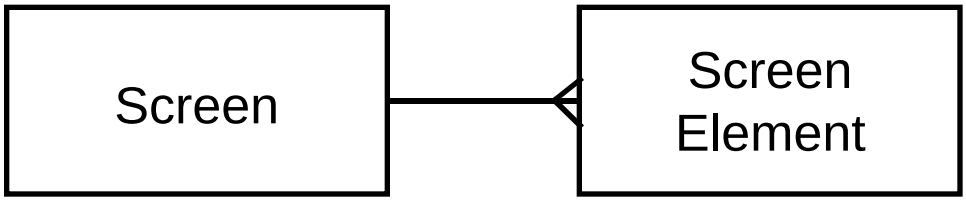


Three Metadata Model Views

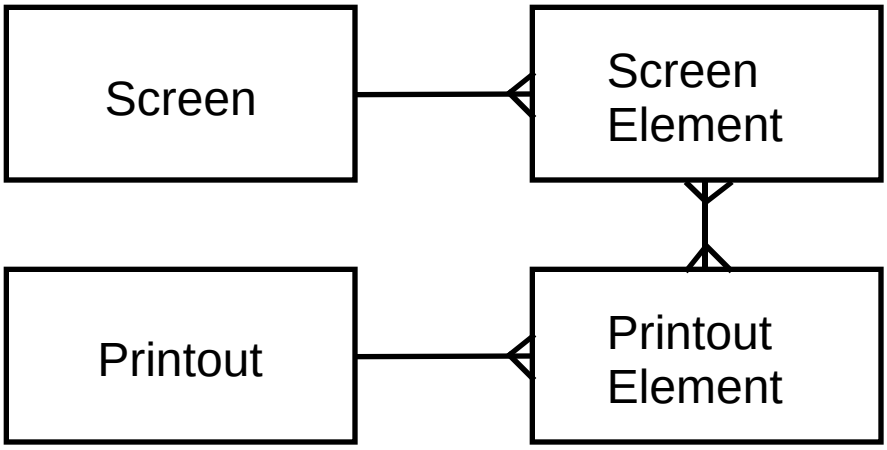




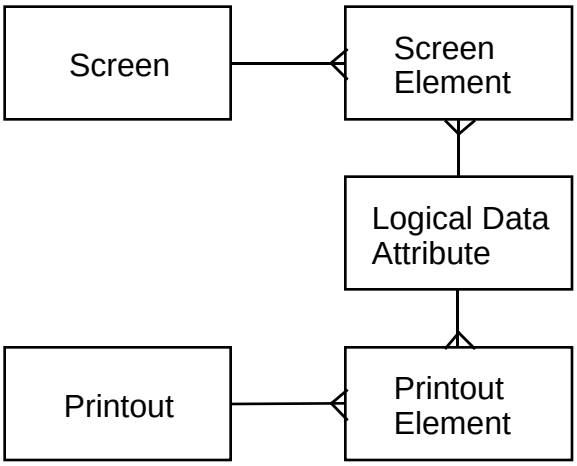
One to many relationship existing between SCREEN and SCREEN ELEMENT



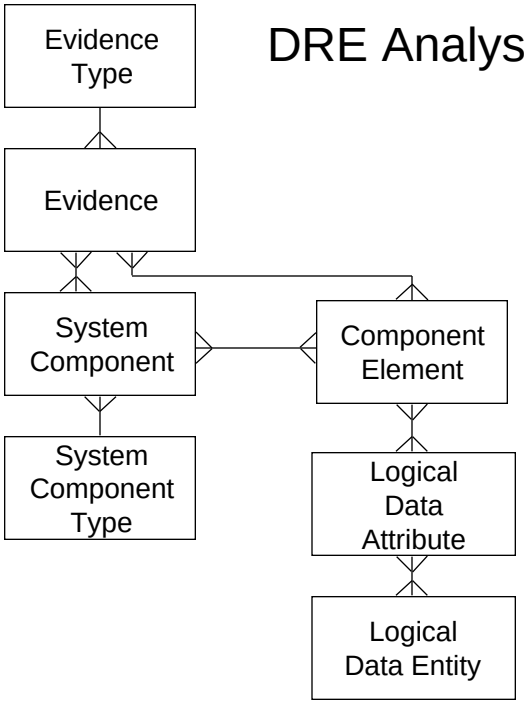
Relationships able to be maintained between metadata components describing instances of the system screens, screen elements, printout elements, and printouts



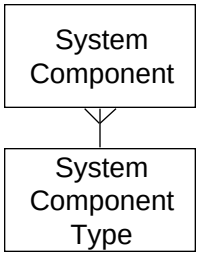
Use of the LOGICAL DATA ATTRIBUTE entity to resolve the many to many relationship between SCREEN ELEMENT and PRINTOUT ELEMENT entities



DRE Analysis Decomposition

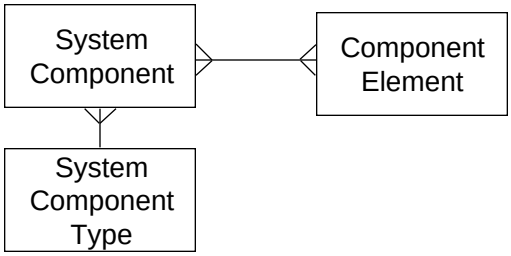


DRE Analysis Decomposition



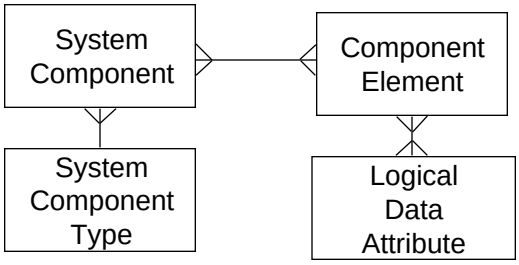
One SYSTEM COMPONENT TYPE is associated with each SYSTEM COMPONENT and many system components can be characterized by each SYSTEM COMPONENT TYPE

DRE Analysis Decomposition



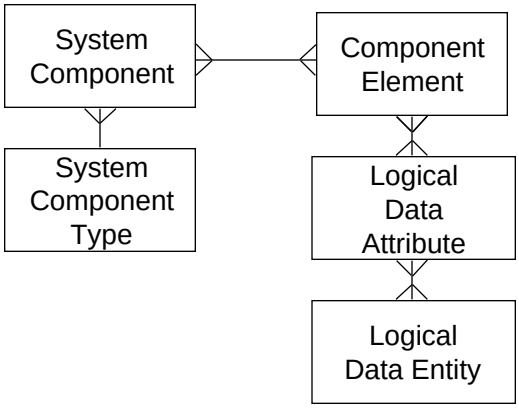
Each SYSTEM COMPONENT can be associated with many component elements and each COMPONENT ELEMENT can be associated with many system components

DRE Analysis Decomposition



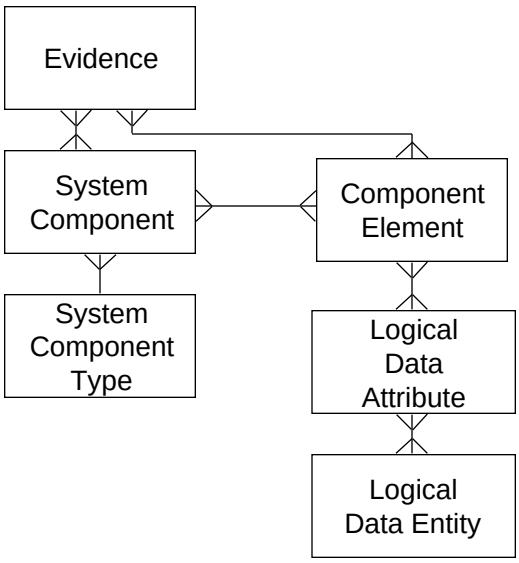
Each COMPONENT ELEMENT can be associated with many logical data attributes and each LOGICAL DATA ATTRIBUTE can be associated with one or more component elements

DRE Analysis Decomposition



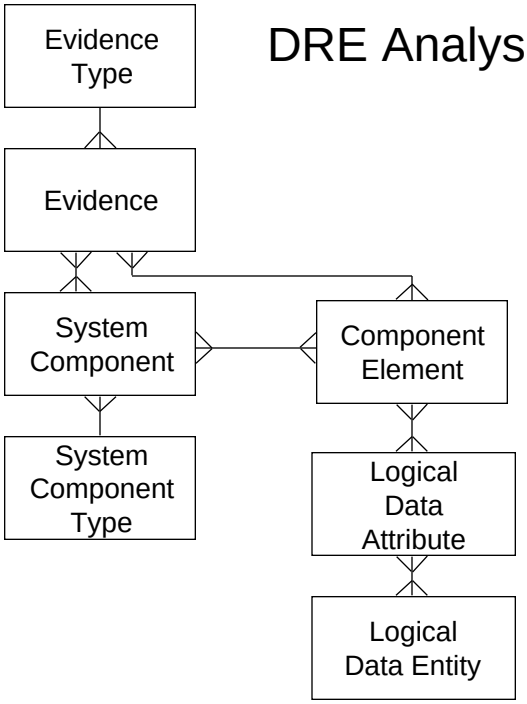
Each LOGICAL DATA ATTRIBUTE can be related to many logical data entities and each LOGICAL DATA ENTITY can be comprised of many logical data attributes

DRE Analysis Decomposition



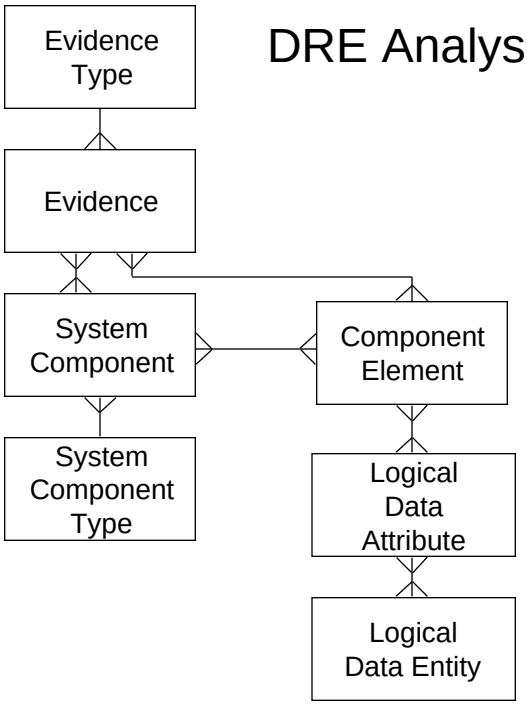
Each SYSTEM COMPONENT and COMPONENT ELEMENT can be linked to multiple pieces of EVIDENCE and each piece of EVIDENCE can be linked to multiple system components and component elements

DRE Analysis Decomposition



Each piece of EVIDENCE can be linked to an EVIDENCE TYPES and each EVIDENCE TYPE can be linked to one or more pieces of EVIDENCE

DRE Analysis Decomposition



Completed DRE Analysis Decomposition

Conceptual Analysis Decomposition

Metadata analysis goal:

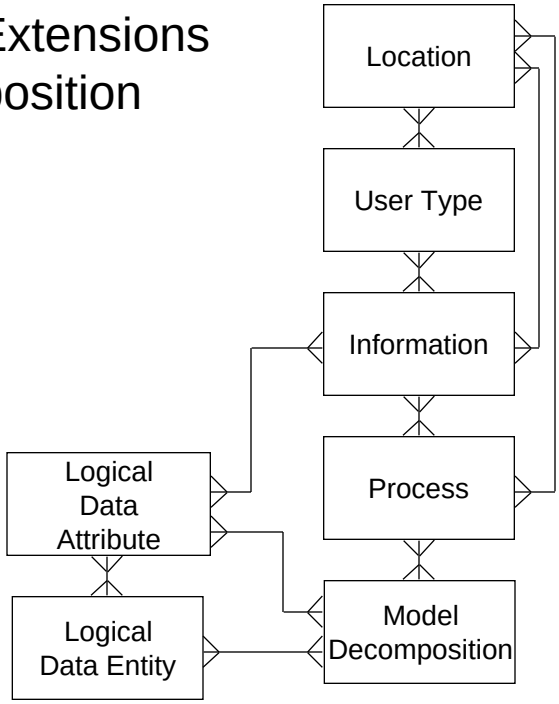
- to link each component element with one and only one specific LOGICAL DATA ATTRIBUTE and ENTITY instance
- achieving common data use through out the model
- permitting the implementation of standard use throughout the system implementation.

Conceptual Analysis Decomposition

When populated you can determine:

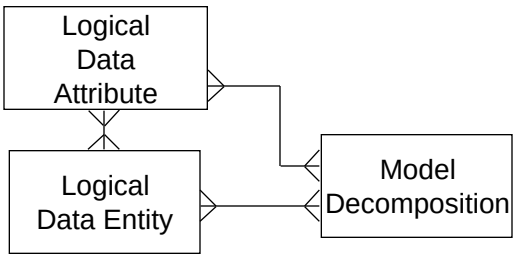
- Screen data attribute X of screen Y is a display of logical data entity Z
- Data entity W is generated by a code location X of job-stream Y that is maintained at location Z
- Printout W is related to LOGICAL DATA ATTRIBUTE X of LOGICAL DATA ENTITY Y
- Therefore it is by definition equivalent to input element Z

Structural Extensions Decomposition



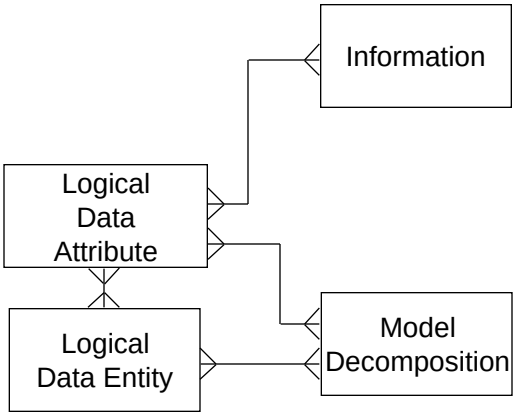
Structural Extensions Decomposition

Each LOGICAL DATA ENTITY and LOGICAL DATA ATTRIBUTE can appear in multiple model decompositions and each MODEL DECOMPOSITION can contain multiple logical data entities and logical data attributes



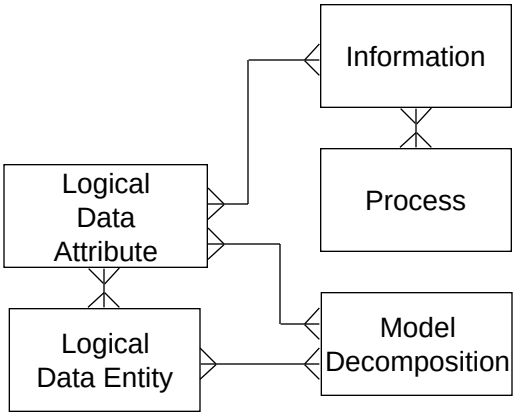
Structural Extensions Decomposition

Each INFORMATION can be composed of multiple logical data attributes and each LOGICAL DATA ATTRIBUTE can be provided as part of many informations



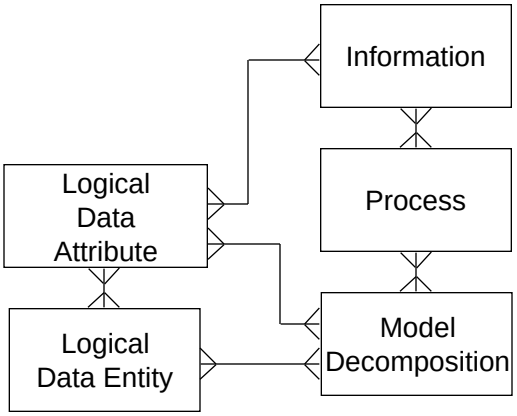
Structural Extensions Decomposition

Each individual INFORMATION can be related to one or more PROCESSES and each PROCESS can require one or more informations



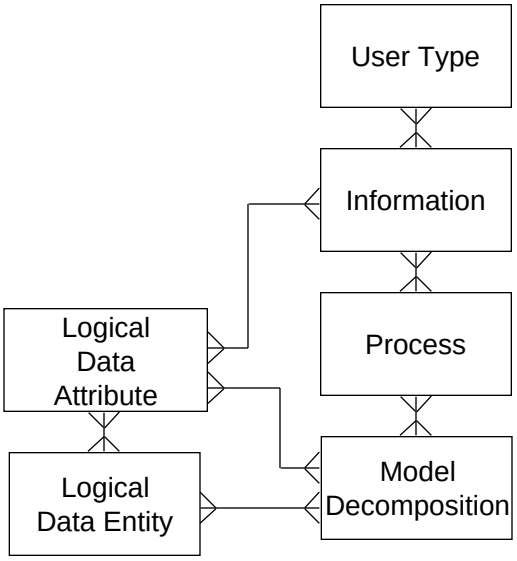
Structural Extensions Decomposition

Each PROCESS can be associated with one or more model decompositions and each MODEL DECOMPOSITION can be associated with one or more processes



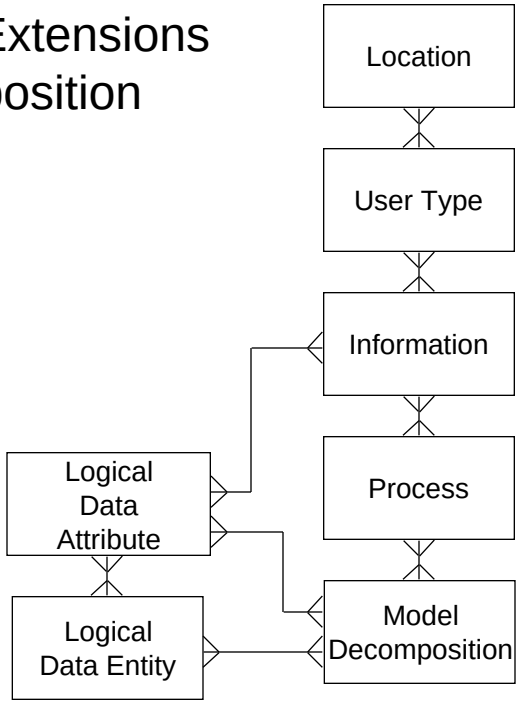
Structural Extensions Decomposition

Each INFORMATION is used by one or more user types and each USER TYPE can request multiple types of informations



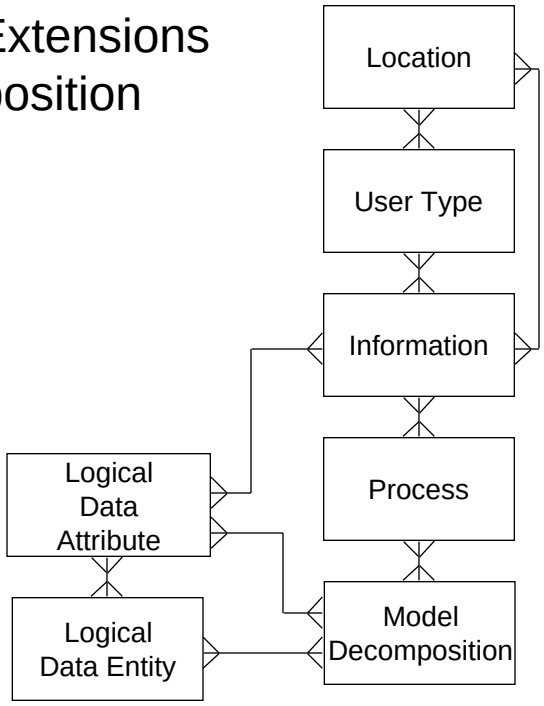
Structural Extensions Decomposition

Each USER TYPE can reside at one or more locations and each LOCATION can have many user types



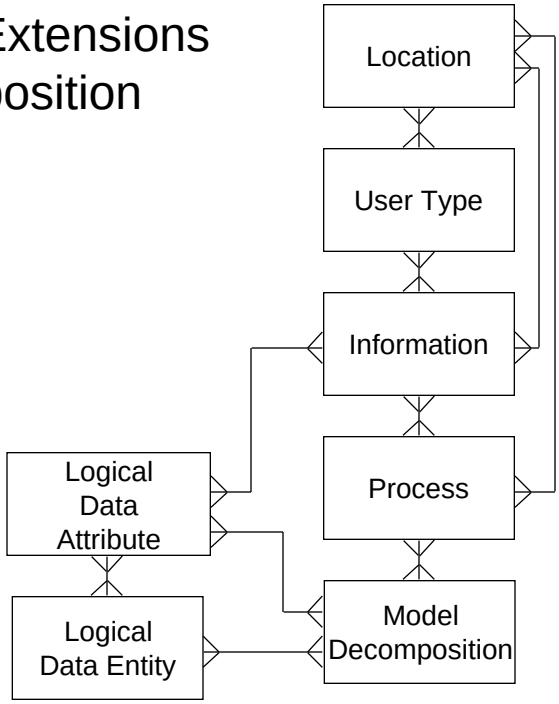
Structural Extensions Decomposition

Each INFORMATION is required by one or more locations and each LOCATION requires one or more informations



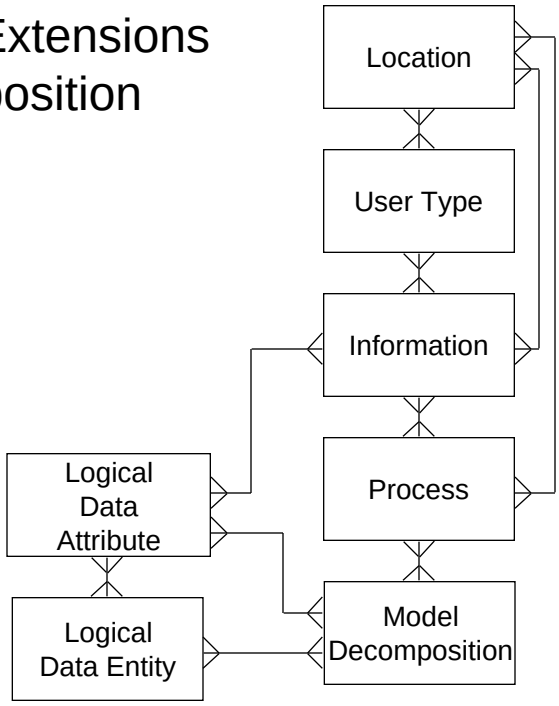
Structural Extensions Decomposition

Each LOCATION may perform many processes and each PROCESS may be performed at many locations



Structural Extensions Decomposition

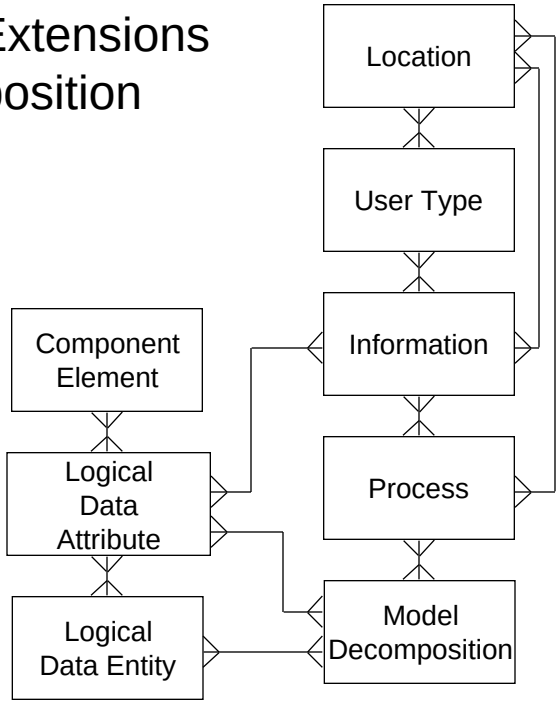
Completed structural extensions decomposition



Structural Extensions Decomposition

Data sharing begins when each COMPONENT ELEMENT is defined as a specific LOGICAL DATA ATTRIBUTE

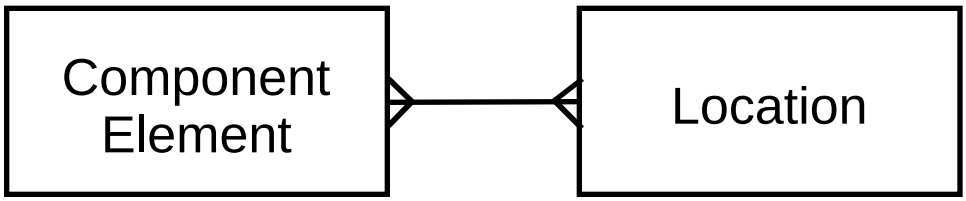
This permits programmatic control over the physical data using logical data manipulations



Contextual Metadata Extensions Decomposition

- It is possible to accomplish data reengineering analysis as a comprehensive examination of the system
- Usually this approach is unnecessary
- Specific metadata entity use is driven by analysis goals
- Certain metadata engineering analyses provide incentives to capture certain additional metadata supporting project specific reengineering objectives
- This information can be represented in the contextual extensions decomposition containing information specifically aiding the analysis at hand
- These may require the addition of specific entities, attributes, or relationships to the metadata model

Example Contextual Metadata Extensions

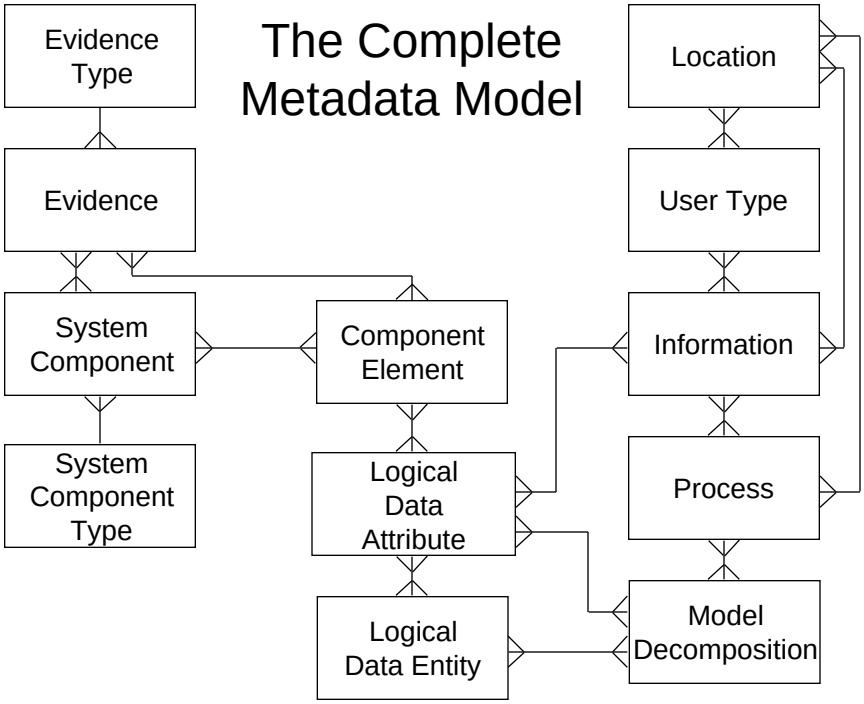


- When working with distributed, unintegrated data, the as is analysis may be aided by linking each LOCATION directly with many component elements and indicating each COMPONENT ELEMENT can be associated with multiple locations

Other Examples

- **Data usage type** - operational or decision support
- **Data residency** - functional, subject, geographic, etc.
- **Data archival type** - continuous, event-discrete, periodic-discrete
- **Data granularity type** - defining the smallest unit of addressable data as an attribute, an entity. or some other unit of measure
- **Data access frequency** - measured by accesses per measurement period.
- **Data access probability** - probability that an individual logical data attribute will be accessed during a processing period
- **Data update probability** - the probability that a logical data attribute will be updated during a processing period
- **Data integration requirements** - the number and possible classes of integration points
- **Data subject types** - the number and possible subject area breakdowns
- **Data location types** - the number and availability of possible node locations
- **Data stewardship** - the business unit charged with maintaining the logical data attribute
- **Data attribute system component of record** - the system component responsible for maintaining the logical data attribute

The Complete Metadata Model



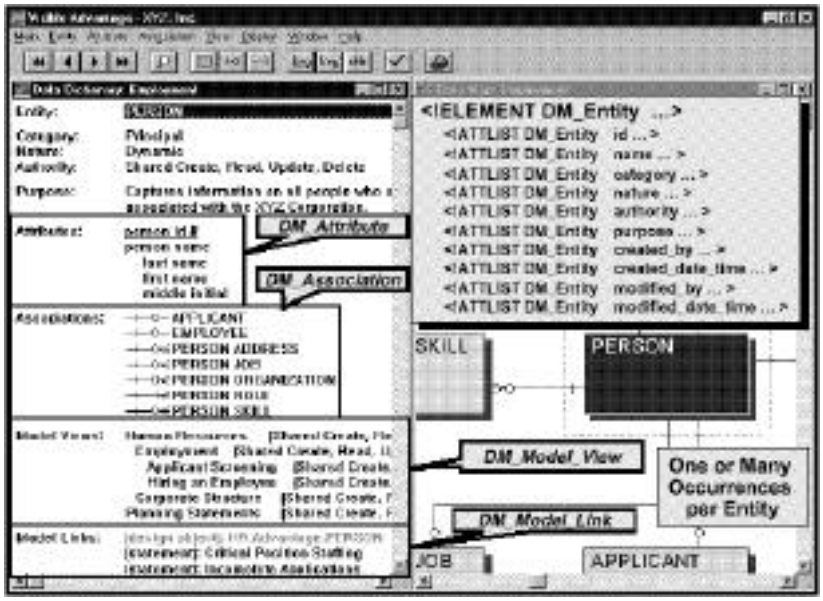
Unresolved Intersecting Entities

- Each SYSTEM COMPONENT can be related to many pieces of EVIDENCE and each piece of EVIDENCE can be related to many system components.
- Each piece of EVIDENCE can be related to many component elements and each COMPONENT ELEMENT can be related to many pieces of EVIDENCE.
- Each LOGICAL DATA ATTRIBUTE can appear in many LOGICAL DATA ENTITIES and each LOGICAL DATA ENTITY is comprised of multiple LOGICAL DATA ATTRIBUTES.
- Each LOGICAL DATA ATTRIBUTE can appear as a component of many informations and each INFORMATION can be comprised of many logical data attributes.
- Each LOGICAL DATA ATTRIBUTE can be associated with many model decompositions and each MODEL DECOMPOSITION can contain many logical data attributes.
- Each PROCESS can be related to many model decompositions and each MODEL DECOMPOSITION can be affiliated with many processes.
- Each PROCESS can be consume many informations and each INFORMATION can be supplied to many processes.
- Each INFORMATION consumed by many USER TYPES and each USER TYPE consumes many INFORMATIONS
- Each USER TYPE can be associated with many locations and each LOCATION can possess many user types.

Extensive use of New Technologies

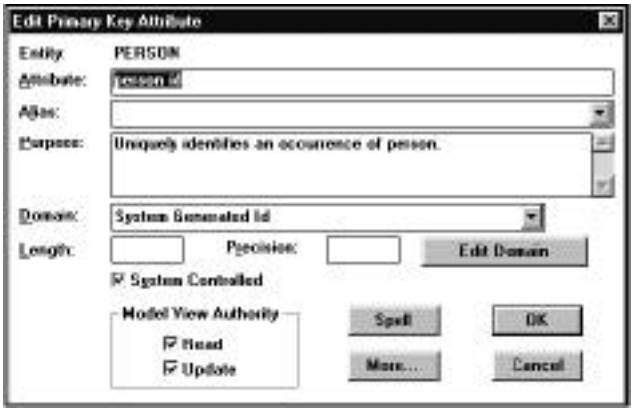
- 4GLs
  - ASP, ColdFusion, Visual Basic
- Data analysis tools
  - Migration Architect
- Business rule engines
  - KPI
  - OST Technologies
  - Business Rule Studio by RuleMachines
  - GURU by Micro Data Base System
- Data logistic networks
  - D2K
- Case Tools
  - Visible Advantage

CASE-based XML Support



http://www.visible.com

CASE-based XML Support

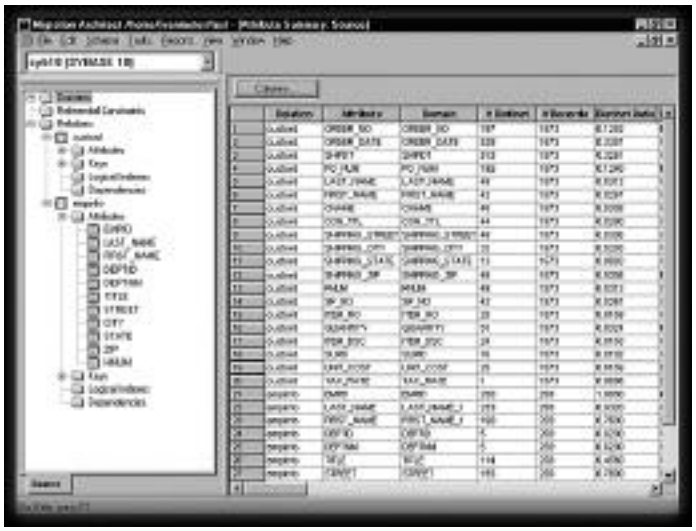


CASE-based XML Support

```
<!ELEMENT DM_Attribute
<!ATTLIST DM_Attribute id            ID                          #REQUIRED>
<!ATTLIST DM_Attribute entity        CDATA                       #REQUIRED>
<!ATTLIST DM_Attribute attribute     CDATA                       #REQUIRED>
<!ATTLIST DM_Attribute type          (Primary-Key|Foreign-Key|
                                     Non-Key | Group)            #REQUIRED>
<!ATTLIST DM_Attribute group         CDATA #IMPLIED >
<!ATTLIST DM_Attribute alias         CDATA #IMPLIED >
<!ATTLIST DM_Attribute purpose       CDATA #IMPLIED>
<!ATTLIST DM_Attribute domain        (Character | Text | Flag |
                                     Money | Numeric | Decimal |
                                     Float | Integer |
                                     System_Generated_Id|Date|
                                     Date_Time|Time)            #REQUIRED>
<!ATTLIST DM_Attribute length        CDATA                       #IMPLIED>
<!ATTLIST DM_Attribute precision     CDATA                       #IMPLIED>
<!ATTLIST DM_Attribute authority     (Read | Update)             #REQUIRED>
```

XML ELEMENT and ATTLIST Declarations for DM _Attribute based on CASE metadata valid values

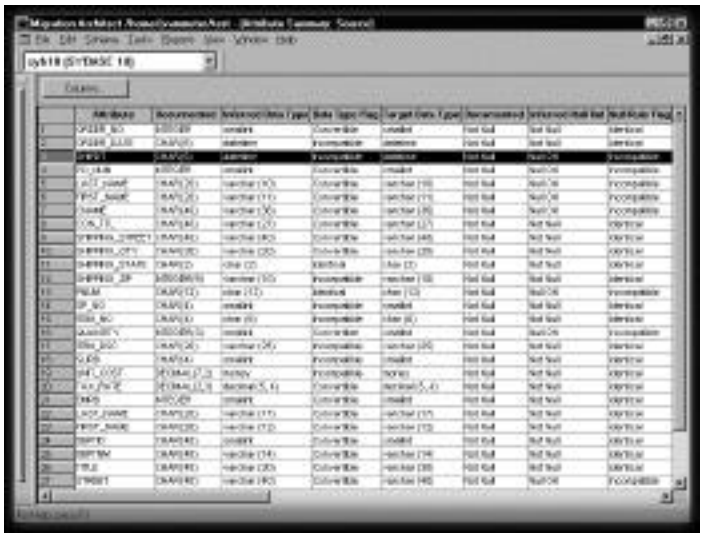
Semi-automating Reverse Engineering: Column Profiling with Attribute Summary Report



Screen shots of Migration Architect used by permission of Evoke Software

http://www.evokesoft.com

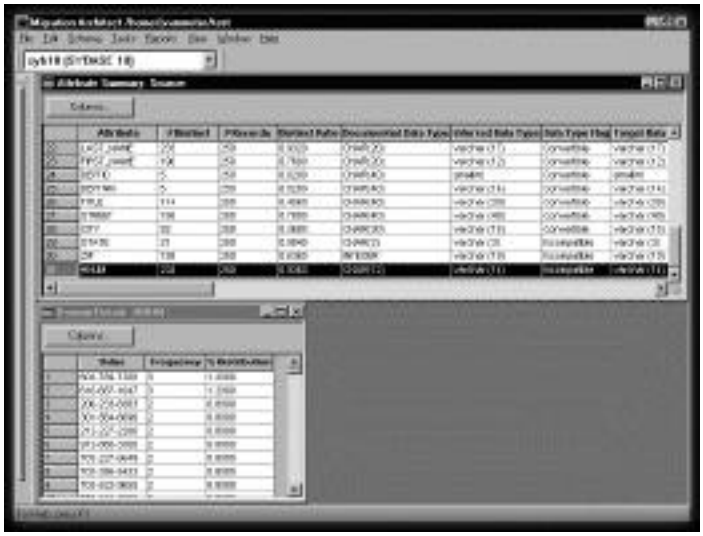
Semi-automating Reverse Engineering: Column Profiling, Compare Documented vs. Actual



Screen shots of Migration Architect used by permission of Evoke Software

http://www.evokesoft.com

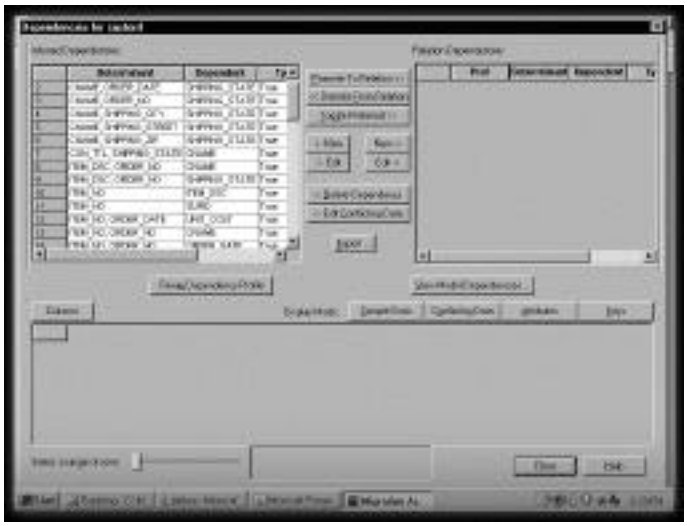
Semi-automating Reverse Engineering: Column Profiling, Drilling Down on Column Values



Screen shots of Migration Architect used by permission of Evoke Software

http://www.evokesoft.com

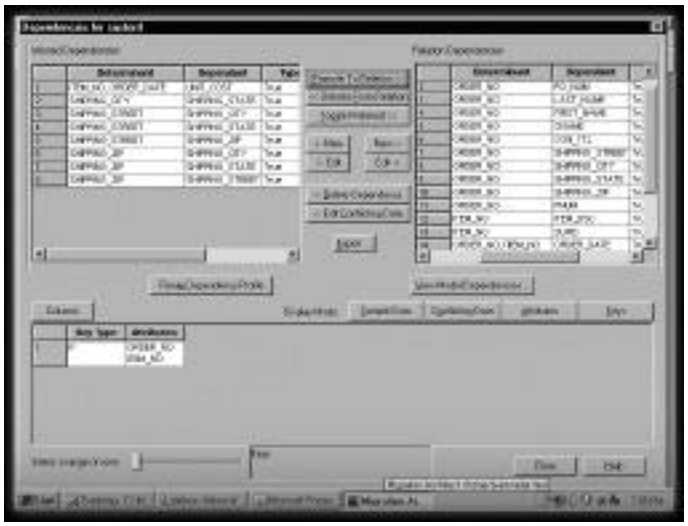
Semi-automating Reverse Engineering: Dependency Profiling, Candidate Dependencies



Screen shots of Migration Architect used by permission of Evoke Software

http://www.evokesoft.com

Semi-automating Reverse Engineering: Dependency Profiling, Promoting Dependencies

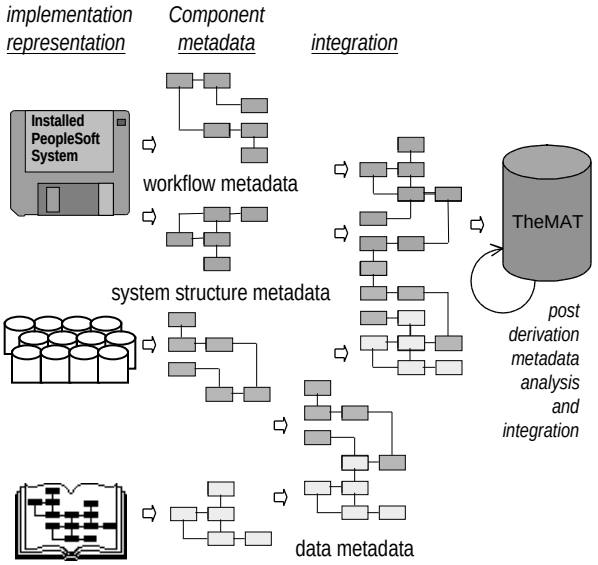


Screen shots of Migration Architect used by permission of Evoke Software

http://www.evokesoft.com

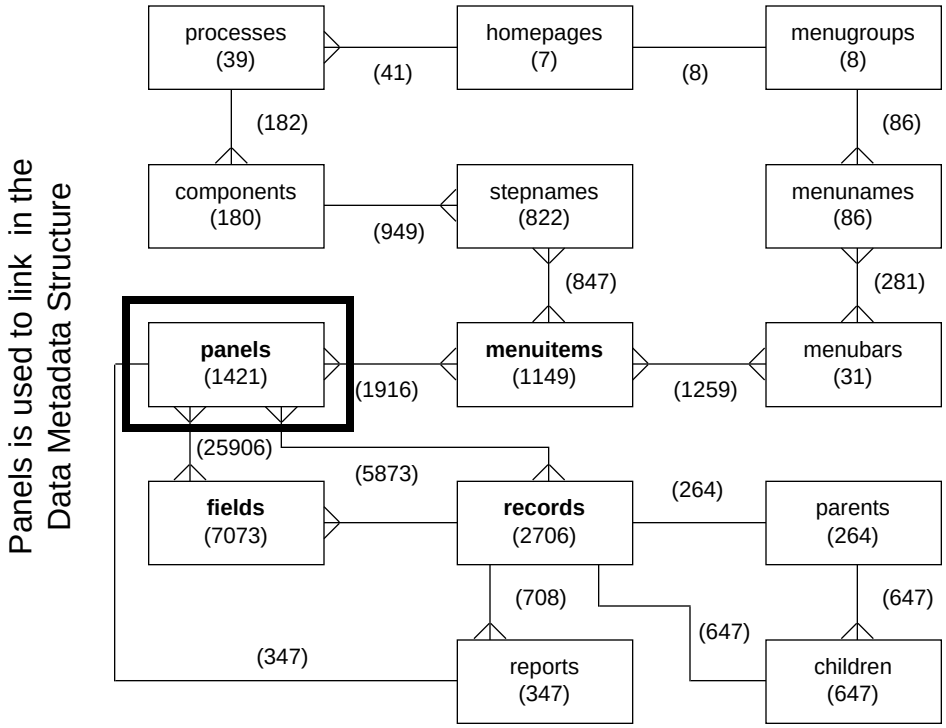
Reverse engineering PeopleSoft

- Queries to PeopleSoft Internals
- PeopleSoft external RDBM Tables
- Printed PeopleSoft Datamodel

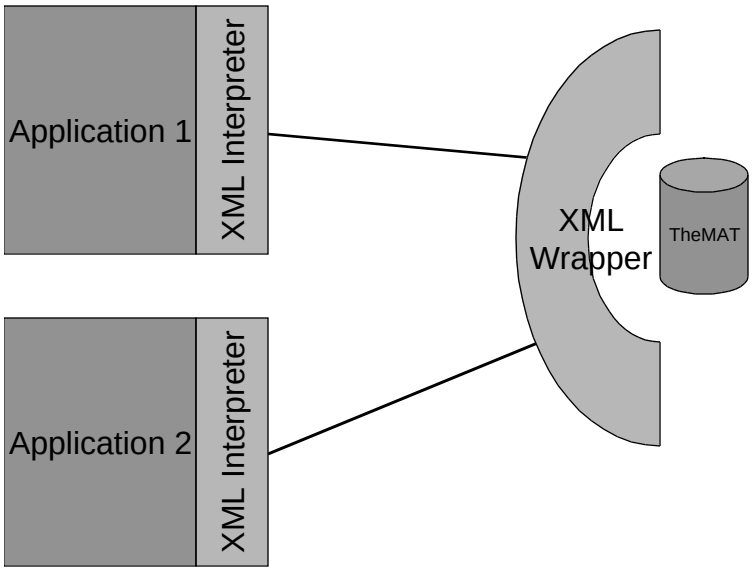


Metadata Uses

- **Workflow Metadata -** business practice analysis and realignment
- **System Structure Metadata -** requirements verification and system change analysis
- **Data Metadata -** data conversion, data security,and user training



Wrapping PeopleSoft Metadata



Metadata Uses

- **Workflow Metadata -** business practice analysis and realignment
- **System Structure Metadata -** requirements verification and system change analysis
- **Data Metadata -** data conversion, data security,and user training

Summary

- Legacy Systems
- Portals Definition
- Development Example
- Metadata Engineering
- XML as a Portal Component
- Advanced Portal Development Technologies
- An ERP-based Example