The convergence of metadata and IT service management

Charles Thomas Betz

A previous version of this work was accepted by the faculty of the Graduate School of the University of Minnesota in partial fulfillment of the requirements for the degree of Master of Science in May, 2003

© 2003 Charles Thomas Betz. All rights reserved.

Table of Contents

1. Abstract

The term metadata is commonly understood as "data about data," but has been expanding and evolving into new domains. This new scope may be seen as essentially the information schema for the enterprise IT function as a whole. The Object Management Group and Distributed Management Task Force have established important standards in this area. Simultaneously, a new overarching discipline for IT operations has emerged, the concept of IT Service Management, along with a set of prescriptive standards known as the Information Technology Information Library. The convergence of these areas is demonstrated and analyzed, along with some possible future directions for alignment.

2. Acknowledgements

Many colleagues have contributed to this work. First, I have gained the acquaintance of many through the dm-discuss email list and the professional organization DAMA International, including Scott Ambler, Dwight Seeley, Alfredo Novoa, Michael Gorman, Terry Moriarty, and Karen Lopez. Noted metadata experts David Marco and Adrienne Tannenbaum also contributed directly and indirectly to this work, as well as Giga analyst Lou Agosta.

Pete Rivett and Sridhar Iyengar of the Object Management Group have helped immensely with my developing understanding of the powerful and abstract OMG metamodeling architecture. Brian Smith of the University of Iowa and my Best Buy colleague Curt Abraham also have provided assistance in this area, as well as participants on the Netbeans and Argo-UML mailing lists.

University of Minnesota faculty Jaideep Srivastava, and John Carlis provided insights, and Gordon Everest of the Carlson School of Management has been generous with his time in helping clarify key points in the evolution of CASE and its associated standards. My advisor Elizabeth Sisley helped with a number of areas, including important clarifications in the areas of configuration management and the SEI's standards.

Credit is also due my supervisors at Target Corporation and Best Buy Corporation for supporting me in this educational goal. Doug Jones at Target Corporation gave me the opportunity to build my first metadata repository, and John Schmidt at Best Buy has presented me with the challenge of integration metadata – a challenge I hope to soon meet! Also at Best Buy, Hugh Juergens provided some provocative thoughts, and Chris Capadouca has never failed to provide explanations of great clarity on the intricacies of ITIL and ITSM.

Many others have contributed as well. Any responsibility for errors or misinterpretations is of course mine alone. Finally, none of it would have been possible without the continued and loving support of my family and especially my wife, Suzanne Magdalene.

Omnes viae Romam ducunt (All roads lead to Rome).

ROMAN PROVERB

…Ice walls flick away like supersonic butterflies made of shade. Beyond them, the matrix's illusion of infinite space…Trying to remind myself that this place and the gulfs beyond are only representations, that we aren't "in" Chrome's computer, but interfaced with it, while the matrix simulator in Bobby's loft generates this illusion . . . The core data begin to emerge, exposed, vulnerable . . . This is the far side of ice, the view of the matrix I've never seen before, the view that fifteen million legitimate console operators see daily and take for granted. The core data tower around us like vertical freight trains, color-coded for access. Bright primaries, impossibly bright in that transparent void, linked by countless horizontals in nursery blues and pinks.

WILLIAM GIBSON, "Burning Chrome"

3. Introduction

This project starts with an intuitive assertion: Enterprise information systems[1] suffer from chronic crises of manageability, integration, and lack of flexibility.

Corporate MIS has been a key enabler of business success for decades now, yet throughout these decades there have always been clear common difficulties across the landscape of large, long-lived IS shops:

---

[1] Enterprise information systems in this context being defined as the large, distributed, data-centric architectures employed by Global 2000 corporations and similarly sized governmental and nonprofit organizations in the course of their daily business. They include custom-written and package systems running on a wide variety of platforms, and more generally the distributed, PC-centric networks that have arisen since the early 1980s.

This paper does not seek a meaningful distinction between information systems (IS) and information technology (IT).

This paper does **not** cover other significant areas of software engineering, such as operating systems, compilers, packaged software, embedded systems, and real-time systems, except incidentally as they may be related to enterprise information systems.

This can be inferred from the IS/IT trade press[2], specialized IT research groups such as Gartner, Giga, and Meta, IT practitioner email lists, and practitioner-focused forums such as the Data Management Association's local and national meetings. Practitioners chronically complain of tangled, undocumented systems that have grown almost organically, and now seem like an impenetrable jungle. Business owners paying for these systems have always asked hard questions about ROI and sought heightened visibility into this jungle, visibility that is surprisingly difficult to achieve. While this questioning perhaps seemed to abate during the dot-com boom, the economic climate of the past 2 years has brought it back sharply.

Some indicative quotes:

> System changes are often documented poorly or not at all, information about structure is rarely documented, conventions for special encoding are often buried in the code, and information on data quality will never be documented. (Olson 2002)
>
> As new interactions between application systems are created, or old integrations are modified, developers will spend many hours trying to find integration touchpoints. The hidden cost to the enterprise will be in longer application development times, lower data quality… and higher operation costs. (Schulte, Blechar et al. 2002)
>
> . . . most organizations lack a consolidated view of all the interfaces and couldn't begin to draw a picture that shows all the connections between their systems. This reality is rampant and the problem is compounded by changes over time. The hairball thrives even in new organizations with modern applications. (Schmidt 2003)

However, it has been surprisingly hard to find empirical or peer-reviewed work on this topic. Academic journals generally focus on the interesting problems of new systems development and emerging technology, rather than the mundane—yet costly! —issues of operations, integration, and maintenance. (There are exceptions of course, such as the IEEE International Conference on Software Maintenance (ICSM).)

The problems of enterprise IT of course might be a life's work. This project proposes to focus on what may be a key element to mitigating these issues: emerging industry standards directly relevant to enterprise IT, in particular the information-centric work of the Object

---

[2] E.g., vendor-supported, guaranteed-circulation magazines such as *CIO Magazine, Information Week, eWeek, InfoWorld, Enterprise Application Integration Journal,* and so forth. This research does not rely solely on these sources, which often frame problems and solutions such that they drive revenue for their advertisers.

Management Group and the Distributed Management Task Force, and the process-centric work represented by the Information Technology Infrastructure Library and the Software Engineering Institute's Capability Maturity Model.

This is a synergistic, integrative work that attempts to contribute new knowledge by closely examining the potential for integration between two major loci of influence in the current world of IT. These areas are:

- Metadata and metamodeling
- IT Service Management and Software Process Maturity

3.1. Description of related work

Metamodeling, as used in this paper, is an imprecisely defined yet important term for information modeling that reflexively targets the information and problems faced by IS/IT and computation generally. It is typically represented in static class models or data models. Significant source work on historic metamodeling and the implementation of systems to manage metadata is to be found in (Narayan 1988; Wertz 1989; Tannenbaum 1994; Tozer 1999; Atkinson, Henderson-Sellers et al. 2000; Marco 2000; Alanen 2002; Tannenbaum 2002). Summaries of and commentary on standards work such as CDIF (Flatscher 1996; Ernst 1997; Bezivin 1998), IRDS (Goldfine and Konig 1988; Wertz 1989; Mylopoulos 2001; Rossiter, Nelson et al. 2001), and PCTE (Thomas 1989) are also covered extensively in the historic sources; ironically, current citations for those source standards are very hard to come by, as they have been superseded by the work of the Object Management Group.[3] Another area of major historical interest is first-generation Computer Assisted Software Engineering (McClure 1989; Spurr and Layzell 1990; Stone 1993; Rosenthal 1998) and its underpinning philosophies, in particular Information Engineering (Finkelstein 1989; Martin 1989a; Martin 1989b; Martin 1989c).

The Object Management Group is the foremost group currently focused on metamodeling as defined for this paper. Significant OMG standards examined here include (OMG 1998;

OMG 2002a; OMG 2002e; OMG 2002c), while related industry books include (Fowler and Scott 2000; Grose, Doney et al. 2002; Poole, Chang et al. 2002; Frankel 2003; Poole, Chang et al. 2003). Academic and practitioner sources specifically addressing OMG work are voluminous; reviewed were. (Crawley, Davis et al. 1997; Bezivin 1998; Bézivin 2001; Bezivin and Ploquin 2001; Eshuis and Wieringa 2001; Bezivin and Gerard 2002; Inmon 2002; Ogbuji 2002; Schulte, Blechar et al. 2002; Stevens 2002)

The work of the Distributed Management Task Force is also an important area of metamodeling. Source and related references include (Sturm and Bumpus 1999; Bumpus 2000; Distributed Management Task Force 2000; Distributed Management Task Force 2002b; Distributed Management Task Force 2002a; Distributed Management Task Force 2002c; Tele-Management Forum 2002; Distributed Management Task Force 2003).

IT Service Management material is to be found in (CCTA 1996; Berkhout, Harrow et al. 2000) with related summary/commentary work from industry sources in (Sturm, Erickson-Harris et al. 2002; Van Bon, Kemmerling et al. 2002).

The work on software process is of course voluminous–e.g. (Boehm 1988; Humphrey 1989; Humphrey 1994; Kan, Basili et al. 1994; Kan, Dull et al. 1994; Kan 1995; Bamberger 1997; Bach 1998; Gray 1998; Rational Software Corporation 1998; Zahran 1998; Hughes, Rosenthal et al. 1999; Jacobson, Booch et al. 1999; Boehm 2000b; Boehm 2000a; Boehm, Port et al. 2000; Holmes 2000; Rajlich and Bennett 2000; Larman 2001; Hartman 2002), with the SEI's Capability Maturity Model (Paulk 1995) serving as a keystone. Other significant related material on IT process is found in the data management and data quality literature, e.g. (Huang, Lee et al. 1999; Brackett 2000; Dama International 2000; Loshin 2001; Redman 2001).

Finally, enterprise architecture is an important related topic that seems to have more practitioner attention, especially via the major research groups, than academic attention. Two well-known sources are (Cook 1996) and (Spewak and Hill 1993). One recent prescriptive

[3] For example, www.cdif.org is no longer a valid URL, and the IRDS standard was officially sunset by ANSI and FIPS.

standard is (The Open Group 2002). One unpublished yet crucial attempt at standardizing a normative information model for this space is (OMG 2002d).

It should be noted that high quality peer-reviewed academic work published in refereed journals was in short supply in all these areas; the few papers by academics that were located were mostly unreviewed workshop material, research group manuscripts, and so forth.

3.2. IT-related standards survey

Standards are nothing new to enterprise IT and computing generally. While often the butt of jokes (*e.g.* "The great thing about standards is there are so many from which to choose") the fact is that industry standards have been **absolutely indispensable** to the evolution of information technology and its social and economic benefits.

These standards have been gradually working their way up a technological "stack," starting with basic electrical power standards and progressing through various physical signaling and circuitry standards, through internetworking and interoperability standards extending directly into software engineering and business semantic domains. It is on the higher parts of the stack that this capstone proposes to focus.

This paper will focus on the efforts of:

- the Object Management Group (UML/CWM/MOF/MDA/etc)
    - Model-Driven Architecture (MDA)
    - Unified Modeling Language (UML)
    - Common Warehouse Metamodel (CWM)
    - Enterprise Distributed Object Computing (EDOC)
    - Meta-Object Facility (MOF)
- Distributed Management Task Force (DMTF)
    - Common Information Model (CIM)
- Information Technology Infrastructure Library (ITIL)

Other relevant standards work that will not be addressed here:

- SEI (CMMI)
- UN/CEFACT and OASIS (ebXML)
- W3C (RDF/Semantic Web)
- COBIT
- the ISO RM/ODP effort
- Business Process Modeling initiative
- The Open Group (Architectural Framework)

In addition to specified standards, there are more loosely defined trends relevant to this discussion:

- IT Service Management
- Enterprise Architecture
- Business Activity Monitoring

3.3. Questions to be answered by this research, or In search of radical transparency

The following questions are often asked in IS shops of any size:

- Who uses this component?[4]
- How does this component contribute to or relate to the business?
- Can I re-use this component?
- What components are dependent on this component?
- What are the consequences of altering this component?
- What are the **business consequences** of problems with this component?

The solutions attempted to these problems have met with at best limited success, and in many if not most large organizations there simply is no efficient way to answer such questions. Determining such questions usually requires massive investments in research. Analysts must often be comfortable with delving into the minutiæ of source code in order to complete such research assignments.

As the application portfolio in a large corporation evolves, it is often the case that new systems are simply accreted onto the old, because the old systems and their dependencies are not well understood. Actually fully retiring and decommissioning old systems (or even

[4] Component in this sense meaning **any** physical or virtual element in the IT domain: server, router, executable file, script, data element, table, directory, etc…

relatively small sub-components) is surprisingly rare in the author's experience.[5] This issue accelerated during the go-go e-commerce world of the 1990s, when the prevailing time-to-market culture infected both internal systems as well as customer-facing (B2C) development. Generally, enterprise IT has always exhibited a highly project-centric mentality, in which larger questions of re-use, impact and meaning are given short shrift, to the long-term detriment of the organization.

Although this problem has been recognized for decades by enterprise IT practitioners, research on the academic side has been limited. Much of the research on this subject is in the area of software maintenance process; the reader is referred for example to the various proceedings of the IEEE International Conference on Software Maintenance (ICSM) (e.g., (De Lucia, Pannella et al. 2001)). However, this field is highly technical in approach and generally does not address the business-oriented issues: how do I understand a given component **in its broader business context?**

The following diagram may help clarify this goal, which might be called "radical transparency":

[5] A search on the word "decommission" across the entire IEEE Computer Society Digital Library resulted in no relevant hits.

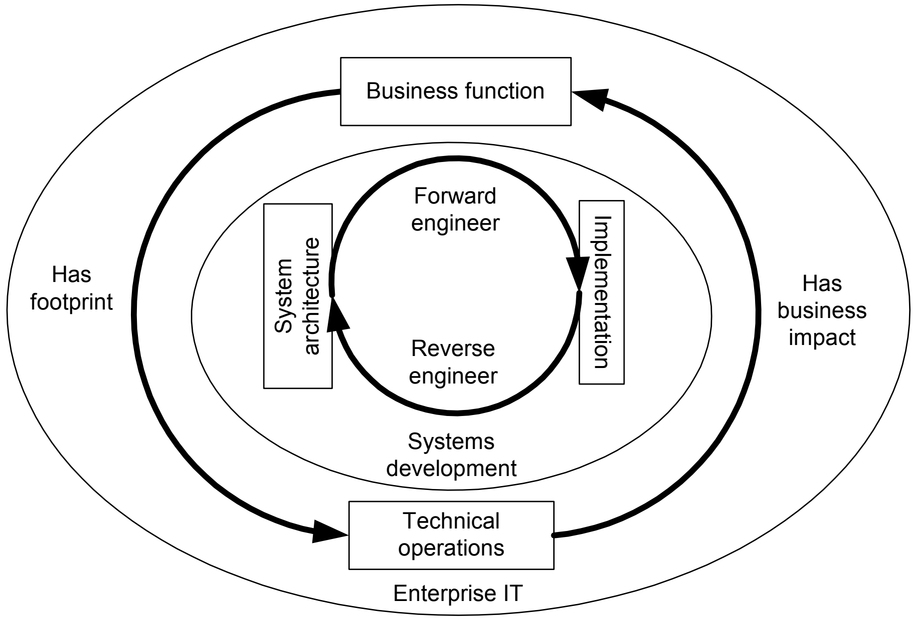

Figure 1. Two round-trips

The round trips referred to are denoted by the heavy round arrows. **The IT crisis can be seen as the inability to navigate these paths without friction.** Radical transparency is precisely this navigability. Metadata is the information needed to traverse these paths.

3.4. All roads lead to Rome

Solutions to the problem of enterprise IT transparency have been sought in several areas, on both the development and operations sides and in the spaces between them:

- Enterprise architecture and IT portfolio management (Spewak and Hill 1993), (OMG 2002d), (Cook 1996)
- Metadata repositories (Tannenbaum 1994; Marco 2000; Tannenbaum 2002)
- Computer-assisted software engineering (McClure 1989; Spurr and Layzell 1990; Stone 1993)
- IT service delivery standards ((UK) 2002) and tools (for example, management frameworks)
- Operations management frameworks (Tivoli, Openview, Unicenter)
- Application reverse engineering and data profiling (Kullbach, Winter et al. 1998; Butler and Clark 1999; Perrochon and Mann 1999; Keller, Bedard et al. 2001)
- and most recently, Business Application Monitoring (Tibco Inc. 2002)

It is interesting to consider these diverse areas in light of the following question:

What is the only major enterprise capability not to have commercially available ERP solutions?

Obviously, enterprise IT itself. A central thesis of this paper is that multiple hitherto separate IT domains are converging into the same general space:

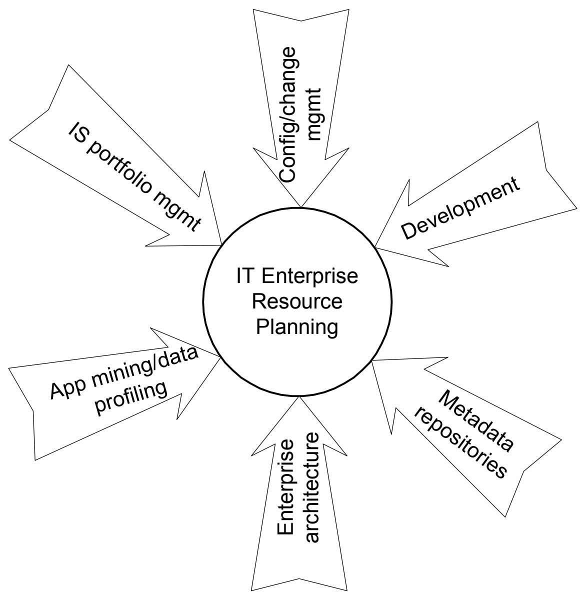

Figure 2. The emergence of ERP for enterprise IT

To succeed in the complex and differentiated domain of enterprise IT, it's a core thesis of this research that such an IT/ERP concept must be standards based and interoperable, and represent a logical evolutionary step beyond the ERP systems available for the other major enterprise domains.

4. Metadata basics

> meta-data
>
> …Data about data. In data processing, meta-data is definitional data that provides information about or documentation of other data managed within an application or environment.
>
> For example, meta-data would document data about data elements or attributes, (name, size, data type, etc) and data about records or data structures (length, fields, columns, etc) and data about data (where it is located, how it is associated, ownership, etc.). Meta-data may include descriptive information about the context, quality and condition, or characteristics of the data…
>
> *The Free On-line Dictionary of Computing, © 1993-2001 Denis Howe*

The concept of **metadata** is perhaps the most widely-used term to describe the general information that IT must manage. One of the earliest located references is (Tsichritzis and Klug 1978):

> The data dictionary/directory is a meta-database: the repository of information about the database. At a minimum, it contains schema and mapping definitions. It may also include usage statistics of various database objects and object types; access and security declarations; control structures for restart; recovery, accounting, and auditing; descriptions of users; and **textual statements relative to the preceding items** [emphasis added].

The term metadata has gradually expanded its scope from the simple data dictionary of the '80s (Narayan 1988; Wertz 1989), to encompass virtually every aspect of IS/IT operations.

A view from the data analysis community still represents the classic core:

"Metadata describes critical elements of data scattered across the organization." (Jahn 2003) In other words, data about data.

This original conception of metadata was well suited to the classic relational paradigm. The core is the data dictionary: a description of data structures and their elements.

Figure 3. The simplest metamodel

However, in other current definitions (Marco 2000), (Tannenbaum 1994), (Tannenbaum 2002), (Inmon 2002), (OMG 2002a) all of the following are considered metadata:[6]

[6] This paper does not address the rich topic of metadata for unstructured/semi-structured information, such as the Dublin Core standard for library collection management.

- Software portfolio (application inventory)
- IT assets (hardware inventory)
- File, database, object, class, and component definitions
- Organizational structure as it relates to IS system control (e.g., data stewardship, business process ownership)
- Data transformations
- Business process documentation
- Batch job operations
- Data quality statistics
- Software configuration management

And beyond these assets come even more problematic requirements. People – in the role of support contacts – appeared as M2-level objects in the Common Warehouse Metamodel, a controversial decision. Project management and even the budget required to run the IT capability could well be linked into metadata, and result in very useful reporting.[7]

Characterized by "instance-of" relationships.

Metadata is thus a difficult term—perhaps even what the political theorists call an "essentially contested concept." As (Huc, Levoir et al. 1997) notes,

> the definition of metadata … does not correspond to any objective reality and [thus can] only be established in relation to different categories of people using these 'objects' and the associated metadata. The definition of metadata would therefore be the subjective reflection of a user community at a given time.

This paper therefore takes a pragmatic approach and defines metadata as:

Data about data,

the systems that manage it,

and the people and processes that create and consume it.

4.1. Metadata domains

To manage this expanded scope, it is helpful to categorize metadata into several overlapping but logically distinct domains, each with different characteristics and audiences.

[7] Credit to my colleague Hugh Juergens for this insight.

Domain	Description	Producer/consumer
Enterprise	High-level, top-down business analysis, carried out at the enterprise level. Not tied to any domain or technology. Parallel decomposition (i.e. Information Engineering) is the basic modeling paradigm.	Business analysts, IS/IT architects working at the enterprise level.
Conceptual	Information and/or process modeling carried out for the purposes of a particular domain, program, or capability. Examples: an enterprise logical data model in E/R format, or an enterprise canonical XML messaging model, OMG "computation-independent models" (CIM). May be behavior or information-centric.	Business analysts, IS/IT architects working at the program level.
Logical	Information and/or process modeling carried out for the purposes of a particular project or initiative, but still at a somewhat platform-independent level. At this level, metadata should be directly traceable to physical implementations. Examples: E/R diagrams elaborated to allow easy forward engineering into DDL; OMG "platform-independent models" (PIM).	Business clients, application architects and developers.
Physical/technical	Actual, real-world implemented information about IS/IT systems. Examples: tables and columns in a database, files on disk, network connections between nodes, data tranformation semantics, OMG "platform-specific models" (PSM).	System administrators, developers

Table 1. Levels of enterprise metadata

Like any map, these levels are somewhat arbitrary, and deviate from the common 3-level conceptual/logical/physical division by creating a new "enterprise" level. This is necessary due to the divergence of high-level conceptual models into multiple domains.

Note that one CASE tool may handle several levels of metadata.

4.2. Metadata architecture

Metadata historically has been focused on the utilization of "scanners," which are specialized interfaces or bridges from metadata sources into a centralized metadata "repository."

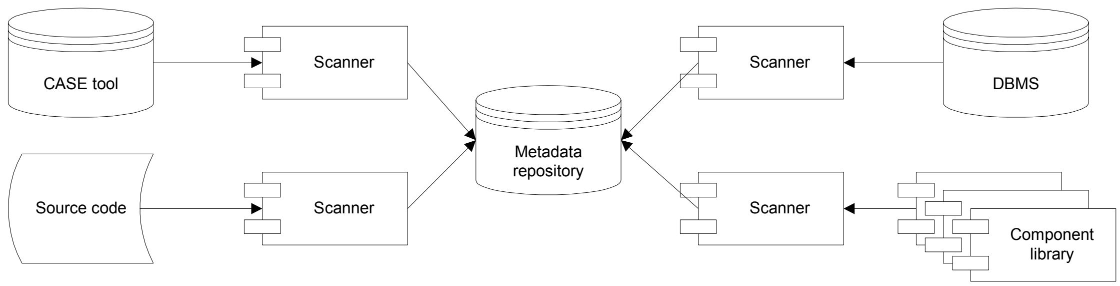

Figure 4. Basic repository architecture

This aggregation into a central repository for integrated reporting is of course very similar to the architectures used in decision support systems.

4.3. Metadata as data warehousing

Comparing the metadata value chain with the regular business intelligence value chain is interesting. BI has historically denormalized operational data replicated into decision support systems (Kimball and Ross 2002). Metadata replicated into specialized metadata management systems has **not** been treated the same way; there is little conception of a metadata "fact" or "dimension," nor would these be generally useful. Where a data warehouse might have a few fact tables surrounded by dimensions, with a 1:100 or orders of magnitude more differential in the dimension: fact cardinalities, the parent: child cardinalities in a metadata repository (in this practitioner's experience) tend to be more on the order of 1:10.

Metadata even in its replicated state is typically highly normalized:

> …several traits that are signposts of highly abstract data models: (1) …more entities than attributes; (2) …more one-one relationships [indicative of subtyped "is-a" relationships] than any other kind of relationship….(Carlis and Maguire 2001)

This high degree of normalization even in the replicated, read-only metadata repository is indicative of how metadata substantially differs from other data problem domains such as DSS.[8]

---

[8] Of course, there is also the Inmon school of data warehousing, which would insist on the existence of a fully normalized enterprise data warehouse from which denormalized data marts are built.

Rather than being used for data analytics (aggregation management, data mining, large-scale decision support, etc) metadata is used primarily for researching **individual** elements (e.g. for impact analysis). This usage pattern, it can be argued, has more in common with the concept of the **operational data store** (Inmon 1999).

And when one is faced with an operational data store (which is inherently an architectural compromise), the immediate question is, should this be transformed into a true, large-scale, integrated system of record?

It must also be noted that metadata is increasingly called on to support the historical view of a given object; the continued expansion of this requirement and corresponding increases in analytics to harvest value from the time series data will start to move the metadata repository more in the direction of data warehousing.[9]

5. Why metadata fails

Despite the long history of, and intensive thought that has gone into the self-referential problem domain of IT metadata, automated management of the internal IT capability itself remains an elusive goal. Large-scale IS/IT continues to be the "barefoot cobbler's child": running its own internal operations with Visio diagrams, Excel spreadsheets, and Access databases, with massive lack of information integrity.

Why is this? I propose three reasons:

5.1. Metadata's scope challenges

Although metadata is clearly technically complex, its large scope perhaps is a more significant reason why so many metadata projects fail: **not for technical, but political reasons.** A centralized IS/IT metadata repository has many of the same characteristics of a centralized ERP system for HR or finance. But while HR or finance represent business

[9] Again, credit to Hugh Juergens for this insight.

processes that are relatively stable, the IS/IT systems development process is still undergoing massive cyclical changes in approach, architectures, and so on.[10]

Besides the severe technical challenges associated with trying to meet many diverse requirements in a single system[11], such a powerful single point of control is bound to also have many political challenges if not supported at the highest levels, and most metadata initiatives have failed to gain such sponsorship. (Often, the organization sponsoring such initiatives is the data administration organization.) Note that sponsorship at the highest level is also typically seen as a requirement for successful business process re-engineering.

5.2. The "system that manages a company's systems?"

(Marco 2003) does make the case that the metadata repository is key to the ongoing operation of the IT capability in the large enterprise, asserting that it is the

> . . . system that manages a company's systems. A meta data repository catalogs all of the applications, data, processes, hardware, software (technical meta data), and business knowledge (business meta data) possessed by an organization.

This assertion however is problematic, when paired with the same author's assertion (Marco 2000) that:

> The process for loading and maintaining the meta data repository should be as automated as possible . . . the task of manually keying in meta data becomes much too time consuming for the meta data repository team . . .

This view is shared by (Poole, Chang et al. 2002) who also call for "eliminat[ing] manual processes" (p. 178). This emphasis irrevocably casts the metadata repository as something **other** than systems that are maintained through manual processes. These systems are critical: they manage the initial population of key metadata generated through processes like asset and configuration management. **Someone** has to be the first one to type in the name of the new server, the definition for a new release of a package, or the name of the new table. What

---

[10] Even supply-chain management, one of the more volatile business domains in recent years, has successful ERP solutions in vendors such as SAP and I2.

[11] Especially if the repository is used for CASE forward engineering as well as documentation.

is that process? [12] And if the metadata administrators are not typing in the information, who is?

The emphasis on automated loading implies that there are operational systems somewhere (modeling tools, management frameworks, source code repositories, etc). And the definition of "manage" starts to come into play here: in this practitioner's experience, the "management" of data is seen as the responsibility of the OLTP systems that capture and update it and provide first-cut operational reporting. The downstream ODS, data warehouse, and data mart systems do not "manage" the operational data; they rather provide value-add correlation and reporting for business intelligence.

In summary, the second reason why metadata fails is that **it is essentially an attempt to build the data warehouse before the operational system is created**, and indeed even before the business processes managed by that operational system have completely stabilized.

5.3. The challenges of integration metadata

Finally, while documenting data about data has proven reasonably straightforward, the next set of questions often asked by IT executives are much harder: **how do things relate to each other?** What processes use this data? Where does it come from, and where is it going to? Few if any IT organizations seem to have solved this issue satisfactorily[13], and some of the issues here will be covered in following sections.

6. The information supply chain problem

Some metadata domains lend themselves well to standard, sophisticated information modeling techniques (class or data model). For example, the classic data dictionary and more

[12] And why do we then assume that the metadata repository is a separate entity (like a data warehouse) from whatever repository is supporting these key IT business processes? Perhaps this makes sense from an operational point of view, but perhaps not. Metadata is small, and to what extent does it need to be replicated?

[13] Among other inquiries, this author asked a plenary of 200 at the 2003 Data Management Association meeting if anyone had integration metadata handled. No hands were raised.

generally the metadata associated with data models can be easily and explicitly reflexively modeled; for extensive examples, see (Tannenbaum 2002) among others. It is also not difficult to define basic deployment metamodels such as the following fragment from the OMG's CWM (which also provides a wealth of data-centric metamodeling):

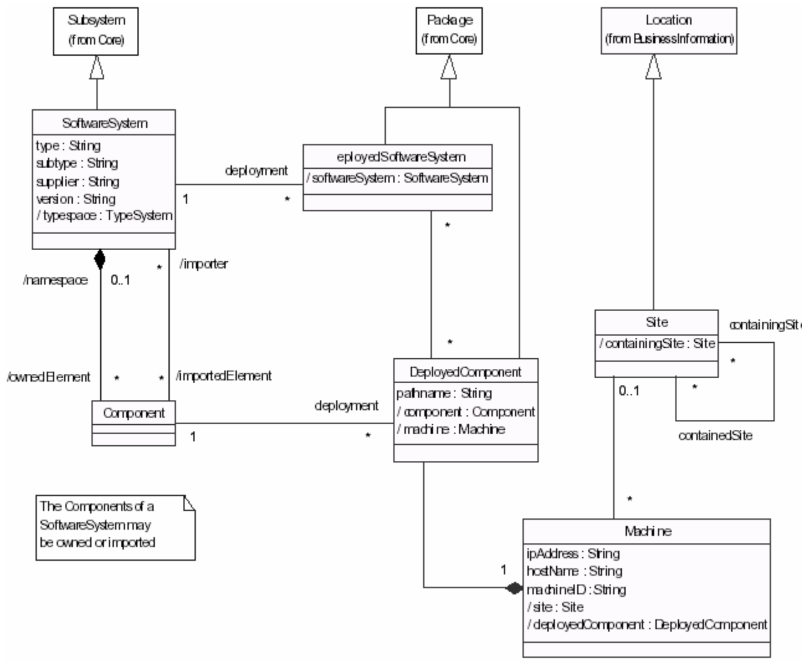

Figure 5. CWM Software Deployment package (OMG 2002a)

This is a perfect example of coherent, understandable metadata that should be reasonably obvious to anyone with a background in entity/relationship or class modeling. However, the information supply chain problem – *aka* the general problem of integration metadata (Schulte, Blechar et al. 2002) is a much more difficult problem.

There is a perhaps unfortunate consensus among modern systems development lifecycle theorists that the venerable data-flow diagram is passé[14]. The reasoning is generally that with relational databases, object-oriented design, and event-driven architectures, there should be no reason to derive and re-derive data. Even metadata theorists subscribe to this point of

[14] The absence of DFDs from the Unified Modeling Language speaks volumes.

view: "The process of deriving information from other information should be discouraged; that is, information should always be derived from data." (Tannenbaum 2002), p. 11.

In this view the architecture is always:

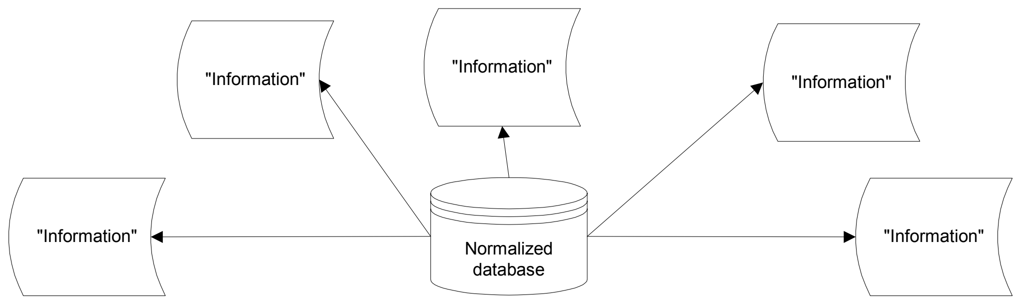

Figure 6. Idealized information flow

The reality is more akin to the good old days of data flow diagramming, as described in classics of structured analysis such as (Yourdon 1975; DeMarco 1979; Page-Jones 1988):

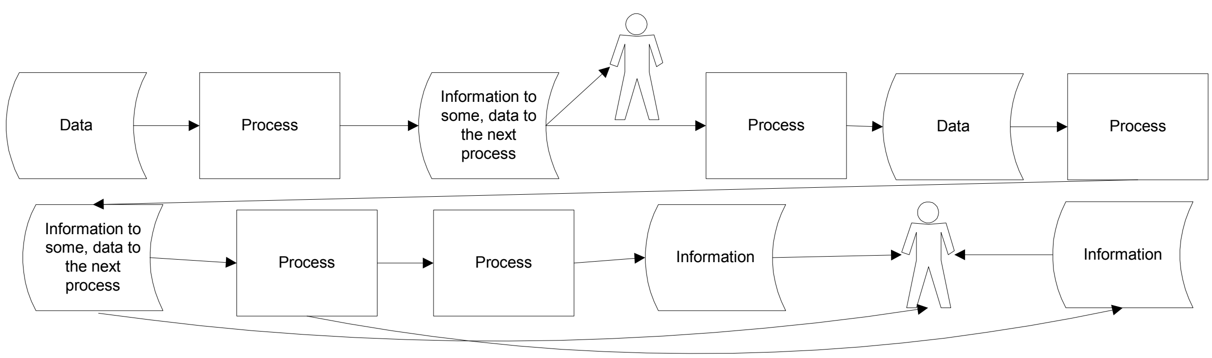

Figure 7. The continuing data-flow reality

This writer believes that large segments of enterprise IT will continue to manifest this design pattern for the foreseeable future. The processing requirements implicit in Figure 6 are just too expensive. This is recognized by (Poole, Chang et al. 2002) in their work on the Common Warehouse Metamodel, which reconciles the OO and data worlds:

> The typical data warehousing and business analysis environment is often described in terms of an Information Supply Chain (ISC)…or information economy…. These metaphors reflect the fact that information in this environment flows from its sources (that is, providers of raw data) through a sequence of refinements that ultimately yields information products that are of great strategic value to corporate decisionmakers.

The information supply chain above essentially is a graph, a concept from discrete mathematics. It can be abstracted thus:

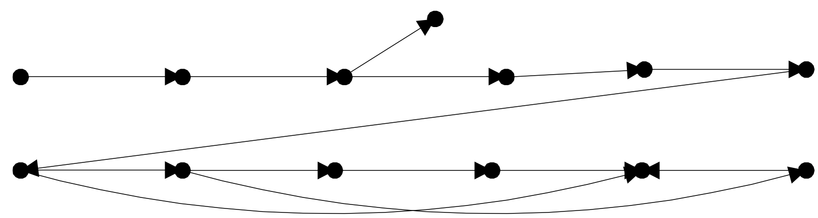

Figure 8. Abstracted Information Supply Chain

The trouble with graph structures is that they do not lend themselves well to relational databases:

- the metamodels required to capture them are abstract
- as (Celko 2000) notes, retrieving graphs "very quickly get[s] into recursive or procedural code."[15]

There is a further complication in that for practical application to IT, the system needs to store graphs and arbitrary paths within them. It is not enough to store the complete transitive closure of an information system as one massive graph:

[15] in the absence of a vendor SQL extension operator such as Oracle's CONNECT BY PRIOR, described in (Celko, 2000).

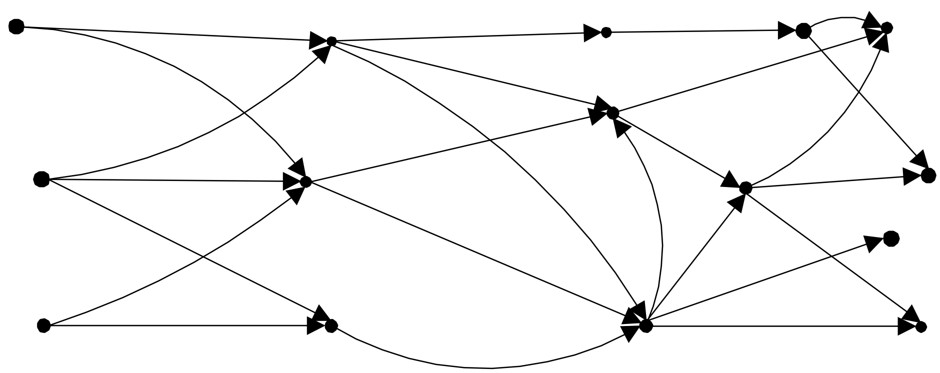

Figure 9 Stored graph

For support, analysis, and general understandability, it is also necessary to store (usually on an opportunistic basis) remembered paths within the integration space.

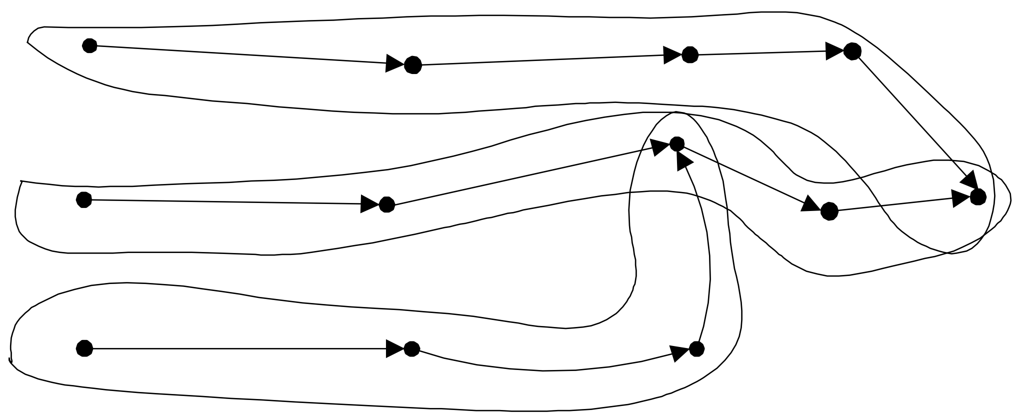

Figure 10 Remembered paths within graph

Consider the formal definition of a graph and a path within a graph:

> A *simple graph* $G = (V, E)$ consists of V, a nonempty set of *vertices,* and E, a set of unordered pairs of distinct elements of V called *edges.*
>
> A *path* of length n from u to v, where n is a positive integer, in an undirected graph is a sequence of edges $e_1, \ldots, e_n$ of the graph such that $f(e_1) = \{x_0, x_1\}, f(e_2) = \{x_1, x_2\}, \ldots, f(e_n) = \{x_{n-1}, x_n\}$ where $x_0 = u$ and $x_n = v$. … (Rosen 1995)

(Carlis and Maguire 2001) cover data structures required for graphs in some detail (p. 347-349). From their treatment, we can see some of the essential problems in storing graphs, subgraphs, and paths in databases. Here is one suggestion adapted from their figure 26-12 (p. 349):

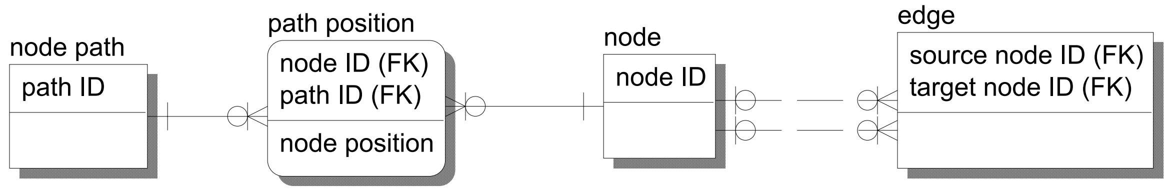

Figure 11. Standard data model pattern for graph and path storage

Let's turn for a time from general theoretical issues of metadata and examine the work of standards bodies concerned with the topic.

7. Metadata standards: The OMG's Meta-Object Facility

> While many of the early CASE tools tried to cover the whole development process, practice has shown that such a generic approach has trouble competing with a series of individual specialised tools. Consequently, CASE tools are becoming more and more open, permitting developers to assemble their favourite development environment from different tools purchased from different vendors yet co-operating via a single interoperability standard. (Demeyer, Ducasse et al. 1999)

7.1. Introduction to the OMG metamodeling architecture

The Object Management Group and associated theorists have been working to build a common semantic infrastructure for defining problem domains using a highly abstract metalanguage called the Meta-Object Facility. Like the preceding IRDS, CDIF, and PCTE standards, this is part of a four-level metamodeling architecture[16]:

[16] These four levels are not related to the four levels of metadata outlined in Table 1. Levels of enterprise metadata on page 18, which would be orthogonal to the OMG standards and basically a horizontal partitioning of the M2 space.

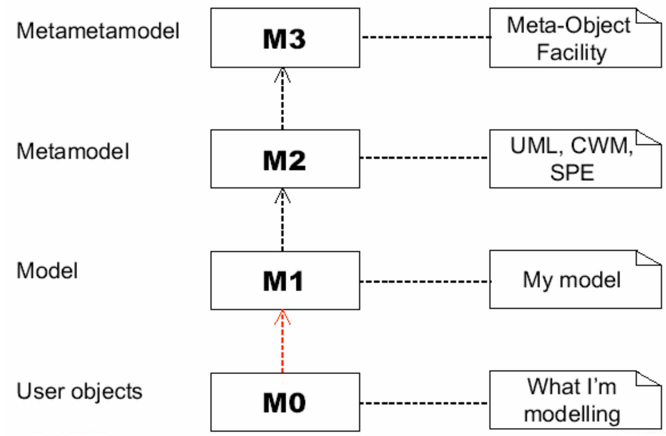

Figure 12. Object Management Group conceptual hierarchy

The M3 level is necessary terse and abstract, just as a modern operating system kernel provides a limited yet extensible set of core services. Although the full MOF encompasses about 30 classes, the specification notes that

> the 4 main modeling concepts are:
>
> 1. Classes, which model MOF metaobjects.
> 2. Associations, which model binary relationships between metaobjects.
> 3. DataTypes, which model other data (e.g., primitive types, external types, etc.).
> 4. Packages, which modularize the models." (OMG 2002c)

These primitives are used to build up a full semantics for defining metamodels. By defining the language for defining problems, the problems can be related to each other.

However, as (Poole, Chang et al. 2002) note,

> Understanding the subtleties of layered abstractions is a challenging intellectual exercise that can be difficult even for experienced professionals. However, it is important to recognize that these concepts, although perhaps difficult to explain, are the foundation of a set of technologies—the OMG metamodel architecture—that are central to the CWM achieving its goal of vendor-neutral metamodel interchange.

7.2. Domains covered

The OMG's core metamodels cover the following areas:

Meta-Object Facility (MOF)	Definition of modeling languages and metamodels
XML Metadata Interchange	Serialization format for M2 and M1 metamodels and models.
Unified Modeling Language	Software-centric modeling of applications
Common Warehouse Metamodel	Data-centric metadata, including relational, flat file, ETL, and related subjects.
Software Process Engineering Metamodel	High-level process model for describing the software lifecycle.
Enterprise Distributed Object Computing	Influential theoretical standard for distributed objects and information flows. Never supported by any major vendors; to be largely subsumed into UML 2.0
UML Profile for EAI	Detailed semantics for capturing application integration data

Table 2. Major OMG infrastructure standards

7.3. Implications of OMG standards for repository architecture

Before the emergence of the OMG standards, the metadata repository vendor owned the scanners:

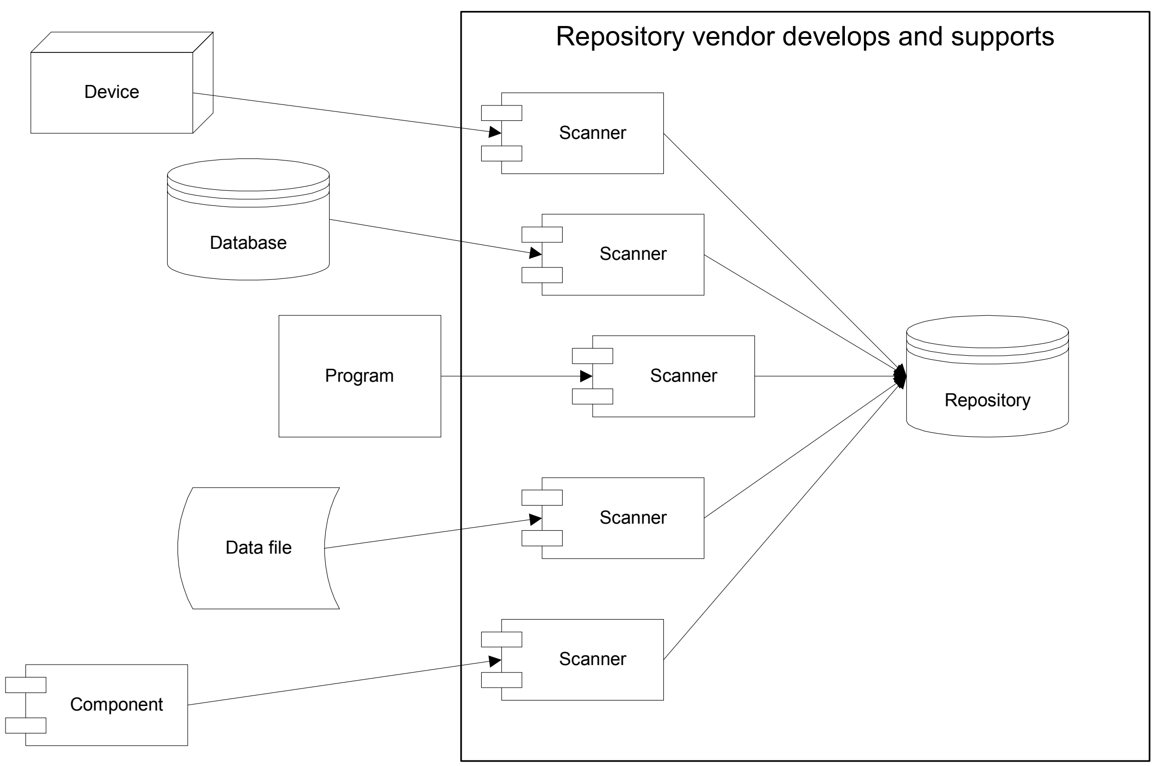

Figure 13. Traditional metadata repository architecture

The list of data sources supported by a previous-generation repository is therefore of keen interest to its potential clients, as the client is **completely dependent** on the vendor to support a given metadata source.

With the move to open standards, the scanner space looks like this:

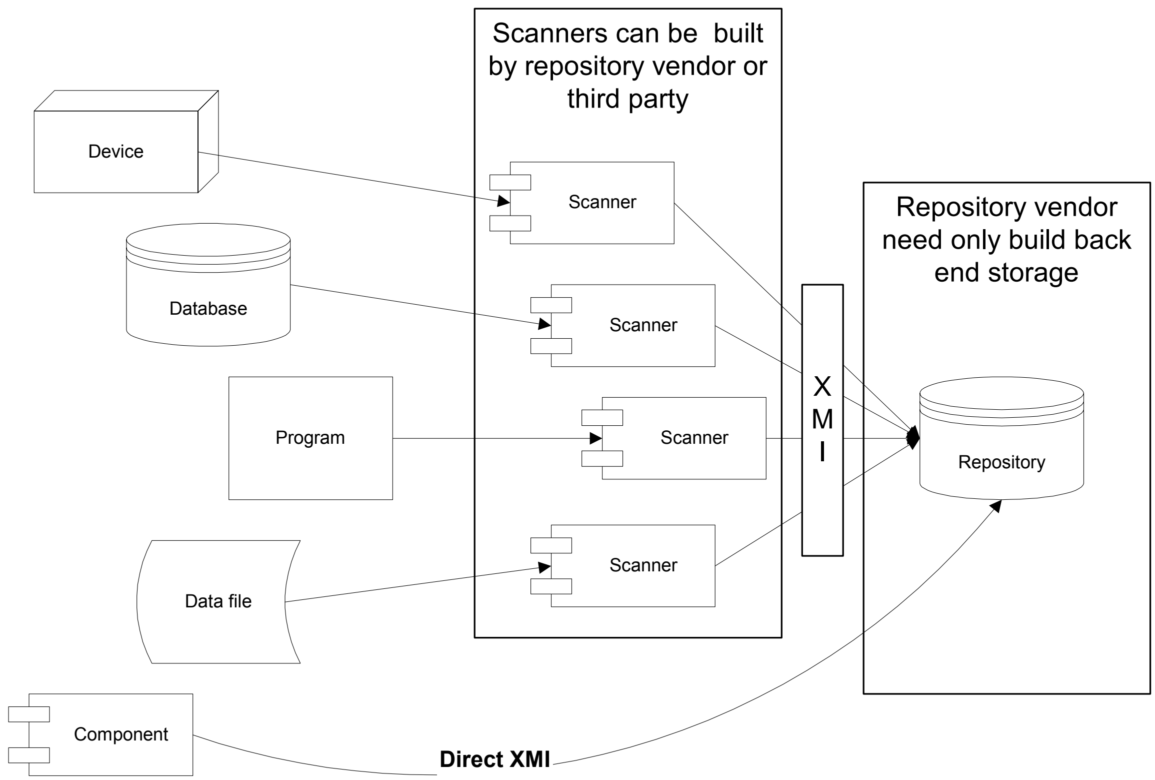

Figure 14. Implications of metadata standards for repository architectures

Firms and even open-source initiatives can specialize and compete in the development of robust scanning technologies, independently of the development and maintenance of the back end repository. **This decoupling provides much greater potential for industry competition and best of breed solutions.** Also, notice that in some cases, the metadata source simply emits OMG-standard XMI as a standard feature (this is true of most UML modeling tools today, and should be encouraged as a key selection factor for all potential metadata sources in enterprise IS).

The consequences of this new paradigm can be seen in a comparative analysis of two major repository products, the CA-Advantage (formerly Platinum) and Adaptive repositories. The number of data sources available for the OMG-based Adaptive Repository already far outstrips those available for the CA-Advantage Repository, which has been in existence much longer.

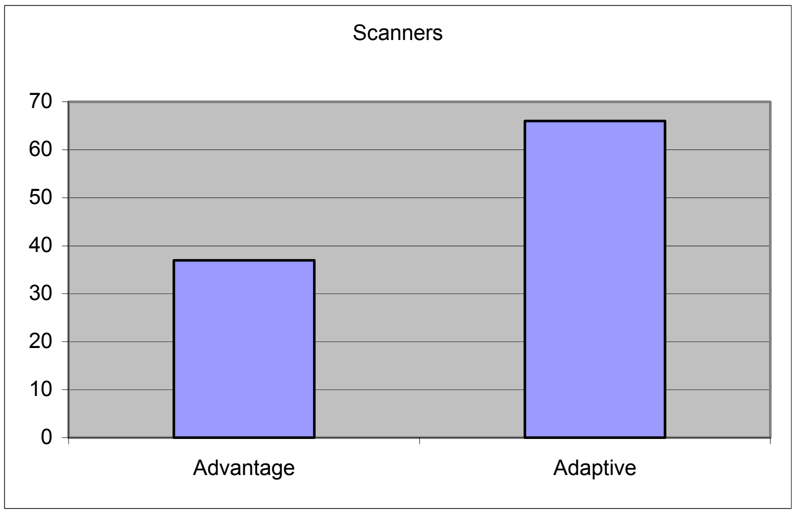

Figure 15. Scanner availability for non-standards and standards-based repositories (original analysis)

The Advantage repository has not seen the development of any new scanners for several years (the Java scanner was apparently the last major development).

8. The Distributed Management Task Force

The Object Management Group does not cover all metadata domains. An important player in defining technical metadata is the Distributed Management Task Force, which has published a series of metamodels covering areas generally lower level than those handled by the OMG (Distributed Management Task Force 2003). However, convergence and overlap is already noticeable. The DMTF work is also interesting in that it provides some integration with the SNMP network management standard.

Application	Details of deploying and managing software products and applications
Core	General foundation elements for describing all DMTF metamodels. Roughly equivalent to OMG MOF.
Database	Database metadata and services.
Device	Many families of physical devices, including detailed decomposition of server components (e.g. processor, cooling, memory, etc).
Event	General system Events and their Indications (occurrences)
Metrics	Historical system information at a granular level, such as transactions, performance, and so forth
Policy	Policies of system performance and operations that in turn support service level management.
Support	IT support and Help Desk activities
System	Hardware, file, operating, systems; processing, logging, diagnostics, etc.
User	Users and security
Network	Network topology, routing, protocols, pipes, filters, SNMP, etc.

Table 3. Major DMTF domains

9. The DMTF and the OMG: opportunities for alignment

Generally, the DMTF and OMG standards inhabit distinct areas in the technology stack:

OMG	Architecture Design Construction
DMTF	Deployment Support Operations

Figure 16. General distinction between OMG and DMTF

However, it is clear that their specifications overlap in some crucial areas and are critically mis-aligned in others. The following sections outline these issues.

9.1. Software systems and deployment

The concept of a software "system" is one of the most critical yet problematic constructs in IT.[17] It is a logical construct, an arbitrary grouping of components associated for a common business purpose.

At level of the Software Development Lifecycle, and IT Service Management, the concept of system is key. Enterprise IT organizations use system names informally to denote logical groupings of components, screens, reports, databases, and so forth that implement particular business functions for the enterprise:

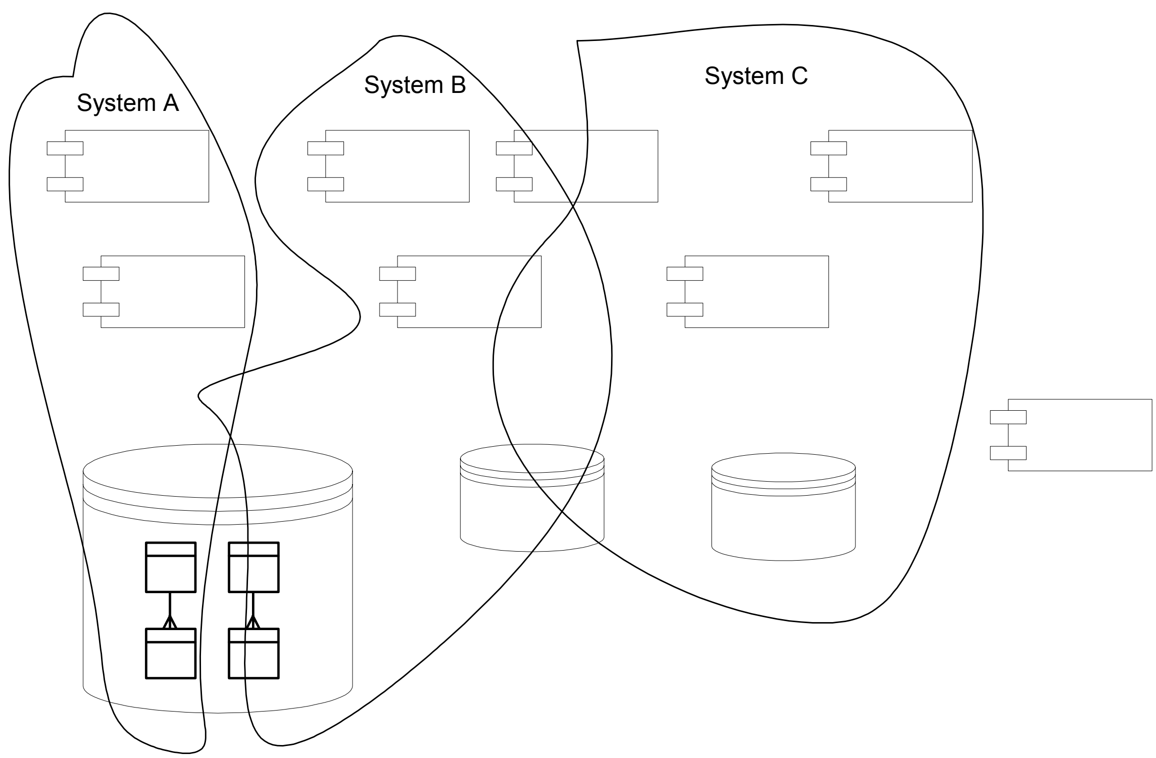

Figure 17 The system problem in enterprise IT

These logical groupings become the key identifiers for help desk, integration, configuration management, and many other IT capabilities. As (Sisley 2003) notes,

[17] A truly rigorous treatment requires a foray into general systems theory, which we will leave out of scope for this paper.

> a global company has hundreds to thousands of such "systems," some owned by IT and many owned by smaller groups in the organization, many with duplicate functionality due to a lack of visibility into what is already owned by the company and could be reused.

The consequences were alluded to at the beginning of this work: “hairball” systems, escalating purchase/build/support costs, and a general lack of manageability that continues to make IT a problematic capability in most enterprises.

One of the key challenges is achieving enterprise agreement on the exact scope and definition of each “system.” **The overlaps and unclaimed components in the above figure are intentional**, and represent a practical challenge for most IT organizations. As Meta analyst (Allegra 2003) notes,

> “fewer than 10% of Global 2000 organizations provid[e] high-level depictions or descriptions of a “systems of systems” view that illustrates the linkage between business processes and information flows to information technology solutions.”

This brings us back to the problem illustrated in Figure 1: how to achieve the radical transparency required for Enterprise IS. I argue that the concept of system is crucial for understanding the outer round-trip, as abstract and imprecise business goals are progressively refined into operational components with support costs, and as the events associated with these components are resolved into impacts on the enterprise’s business goals.

The OMG’s definition of system is

> A collection of connected units that are organized to accomplish a specific purpose. A system can be described by one or more models, possibly from different viewpoints. (OMG 2002a)

This is similar to the DMTF definition:

> The definition of "System" in the CIM context is quite broad, ranging from computer systems and dedicated devices, to application systems and network domains. (Distributed Management Task Force 2000)

The interesting thing with the actual OMG system metamodels is that, despite the generality of the OMG definition, they are all **software** system representations. The OMG has proposed several models for describing software systems and their components.

1. The Common Warehouse Metamodel Software Deployment package (OMG 2002a)
2. UML Deployment and Component diagrams (OMG 2002g)
3. The draft Software Portfolio Management Facility (OMG 2002d)

While none of these specifications go into the detail that the DMTF does on deployment particulars, they provide critical semantics for describing application design and interaction. The Common Warehouse Metamodel's Software Deployment Package explicitly "attempts to capture only as much of the operational configurations as is needed to service other CWM packages; it does not try to be a complete or general-purpose model of any data processing configuration." (Poole, Chang et al. 2002) p. 102.

The DMTF usage of system, on the other hand, is much more focused on hardware systems, generally represented in the System schema of the Common Information Model.

In order to further align the OMG and DMTF worlds, consensus on the exact semantics of "system" and its hardware- and software-centric variations will be critical. This may require an excursion into general systems theory this researcher is unwilling to undertake at this time.

9.2. Batch operations

Both the DMTF's CIM System schema and the Warehouse Process/Operations packages of the OMG's Common Warehouse Metamodel contain models for describing batch processing; however the DMTF's appear to be more general, while the OMG's are specific to data warehousing.

9.3. Core mis-alignment

One major issue with the DMTF is that they have adopted UML syntax without adopting UML semantics:

> The Common Information Model (CIM), which models concepts related to the management of technological assets, is expressed in a UML that would probably be entirely alien to most developers. (Ogbuji 2002)

> Unfortunately, the DMTF CIM is *not* based on UML; rather, it is based on the DMTF MOF (Managed Object Format) language. Specifically, there are many conflicts between the DMTF MOF and standard UML. (Tele-Management Forum 2002)

The DMTF has adopted its own human-readable persistence format, also with the acronym MOF (for Managed Object Format), and its own XML-based serialization format (Distributed Management Task Force 2002b). There is clearly substantial room for further DMTF/OMG alignment.

9.4. Competing architectures

Finally, the DMTF Data schema (and its associated SNMP MIB) pose an interesting implementation conundrum when contrasted with the OMG's CWM Relational package.

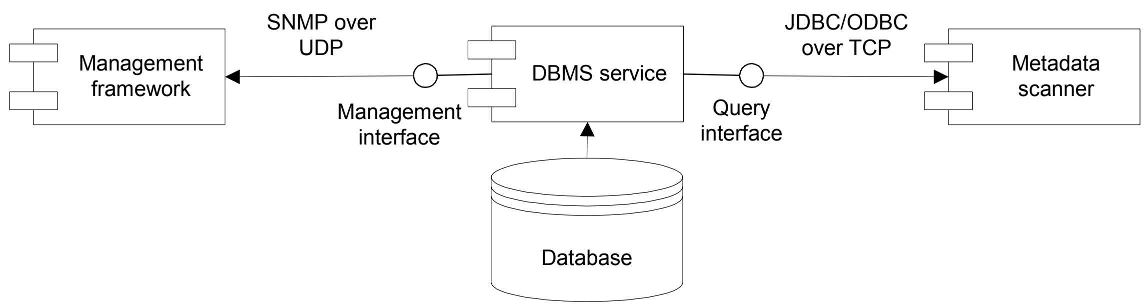

Figure 18. Alternative metadata extraction approaches

Similar descriptive data can be harvested from the DBMS via two paths, SNMP and JDBC/ODBC. This is the clearest illustration of the convergence of the OMG and DMTF domains. While the current OMG and DMTF data specifications may have minimal overlap at this date, the direction of the DMTF is to "Extend the model to cover relational database content: SQL, Relational schema objects…[and] Leverag[e] other modeling efforts, such as OMG's CWM meta-model in the database warehouse domain" (Distributed Management Task Force 2002a), a direction clearly leading to substantial overlap with existing OMG work.

10. IT process

The "ERP for IS" line of thought leads us to the logical question, what are the business capabilities and processes that act upon metadata? **It is a truism in information systems**

that one needs both process and data. The metadata repository and management data schema conversation to date has focused primarily on the metamodels involved; the **static** information structures. To the extent that process is addressed at all by authors like Marco and Tannenbaum, it more or less assumes that the metadata repository is a passive aggregation, fed by batch cycles from often unspecified sources – in keeping with the above comparison of metadata to an operational data store or data warehouse.

This is starting to change: Consider the following passage from (Poole, Chang et al. 2002)

> The meta data management strategy determines, amongst other things, the overall *workflow* of meta data throughout the environment. For example, the meta data management strategy generally determines which products or tools are responsible for authoring meta data; which are responsible for publishing, staging, updating, and managing existing meta data; and which are read-only consumers of shared meta data. (p. 35)

Again from Poole:

> CWM does not define a strategy for meta data management. A coherent meta data strategy is key to the successful application of any meta data integration solution, whether CWM-based or not. An overall meta data management strategy, however, is not defined by the CWM.

And finally:

> The meta data management strategy must identify all possible system components that may serve as either producers, consumers, or both of metadata. (p. 173).

This sounds like a call to process analysis and improvement. But if one focuses on the processes involved, one is immediately confronted with the question, "what does the well-run IT capability look like?"

For example, (Poole, Chang et al. 2002) present a "Sequence diagram for creating a relational table." This illustration is presented at the programmatic level, and looks something like this:

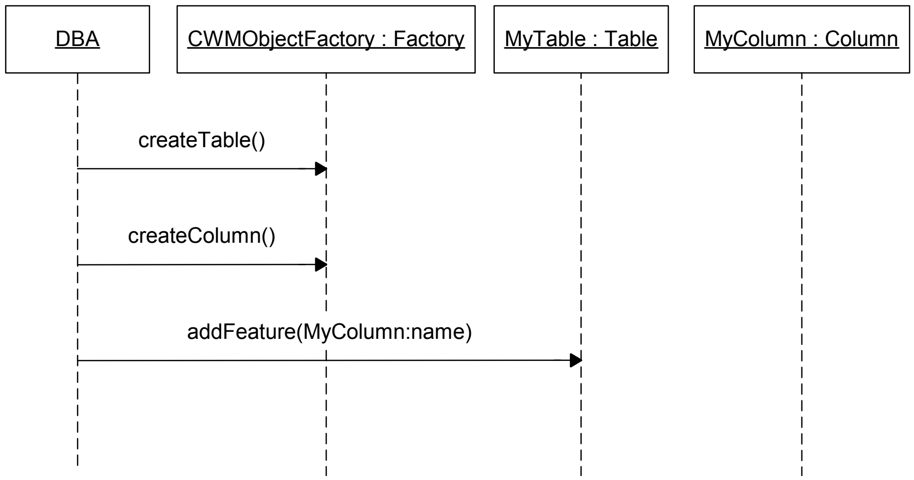

Figure 19. Excerpt from CWM sequence diagram

Adapted from (Poole, Chang et al. 2002)

This is a highly revealing illustration when examined from a perspective the authors perhaps did not intend. Consider the following questions:

- Who is a "DBA"?
- What is the DBA's authority to create Tables and Columns?
- Where does the DBA fit in the functional architecture of the enterprise?
- What are the functional inputs or preconditions to creating tables and columns?
- Who are the consumers of the new data structures that have been created? What processes does this enable?

10.1. Process improvement

> While process improvement may have impacted many other areas of an organization, Information Systems is one of the last areas in business to implement these concepts, even though it may have the most to gain. (Cassidy and Guggenberger 2001)

The concept of process improvement is well-established in manufacturing, engineering, and other major sectors of industry. Methodologies such as TQM and Six Sigma are widely used to continuously improve a wide variety of organizations and processes. The application of

TQM-like disciplines to the field of IT has been primarily in the area of software development, led by the SEI's Capability Maturity Model (Paulk 1995).[18]

While the SEI has branched out into "software, people, and software acquisition, and assisted in the development of CMMs for Systems Engineering and Integrated Product Development," (SEI 2003), the main focus of the SEI's efforts remains the **construction and implementation** of systems, products, and software. Ongoing operations is generally not addressed by SEI research or standards.[19]

As (Van Bon, Kemmerling et al. 2002) note,

> An IT application . . . only contributes to realizing corporate objectives if the system is available to users and . . . supported by maintenance and operations. **In the overall life cycle of IT products, the operations phase amounts to about 70 to 80% of the overall time and cost. . .** p. 29, emphasis added.

This operations phase of the life cycle is the domain of IT Service Management.

10.2. Overview of IT Service Management

Metadata historically has been concerned with a very static, data-focused portrayal of its subject matter. Clearly, to "manage a company's systems," one also needs **processes**.

How are

- infrastructure components deployed?
- data structures and elements designed and approved?
- software systems designed, built, and installed?
- systems monitored?
- problems resolved?
- capacity requirements planned for?

These types of operational IT concerns have historically not been part of metadata scope, at least until recently, and as has been demonstrated above, metadata is not sufficient to solve

[18] This is an exhaustively covered area and will not be summarized or cited in this paper; reader familiarity is assumed.

[19] Which is why CMM/ITIL alignment is such a hot topic in the ITSMF and related bodies.

them. Management frameworks, along with a variety of point solutions used in the data center, IT's own back office, and on the help desk, have filled these requirements.

Even today, the metadata concept is missing some key components to be of much relevance to CIOs seeking to run their internal IT shops. At best, metadata is an after-the-fact data warehouse or ODS-like capability that integrates from sources like CASE tools, operational systems, and others. This missing dimension appears to be emerging in a set of operational disciplines known collectively as IT Service Management (ITSM).

10.2.1. What is ITIL?

Most IT professionals have probably heard of "change management," "configuration management," "asset management," "problem management," "incident management," and the like. The precise meaning of many of these terms can be non-obvious (e.g., what is the difference between problem management and incident management?).

The Information Technology Infrastructure Library, or ITIL (Berkhout, Harrow et al. 2000) is a UK-based set of government standards that precisely define the scope for these terms:

- Help desk
- Software control & distribution
- Contingency planning
- Service level management
- Configuration management
- Change management
- Incident management
- Problem management
- Capacity management
- Application management
- Data management
- Availability management
- Cost management

ITIL is perhaps the closest thing ITSM has to a canonical, authoritative representation. It has been around since 1988, and is increasingly influential in enterprise IT (Hayes 2002). The IT Service Management Forum is the leading organization; its first U.S. conference drew 300; the second drew 800. Expectations are high for the third U.S. conference this fall. (IT Service Management Forum)

In Minnesota, most of the large IT organizations are examining the ITSM discipline and several are moving towards aggressive implementation. Summarizing the entire scope of

ITSM and the ITIL standard is out of scope for this research, which will focus on an examination of the data infrastructure that underlies ITSM processes.

10.2.2. Configuration Management

A core ITSM discipline is the concept of Configuration Management. As (Van Bon, Kemmerling et al. 2002) note,

> "Configuration Management addresses the control of a changing IT infrastructure (standardization and status monitoring), identifying the configuration items (inventory, mutual links, verification, and registration), collecting and managing documentation about the IT infrastructure and providing information about the IT infrastructure to all other processes."

The ITIL material is quite practical and emphasizes the need for tooling to support these capabilities. Central to the entire structure is the concept of a "Configuration Management Database." A CMDB, according to ITIL, manages hardware, software (custom and package), data feeds, databases, documentation, and much more.

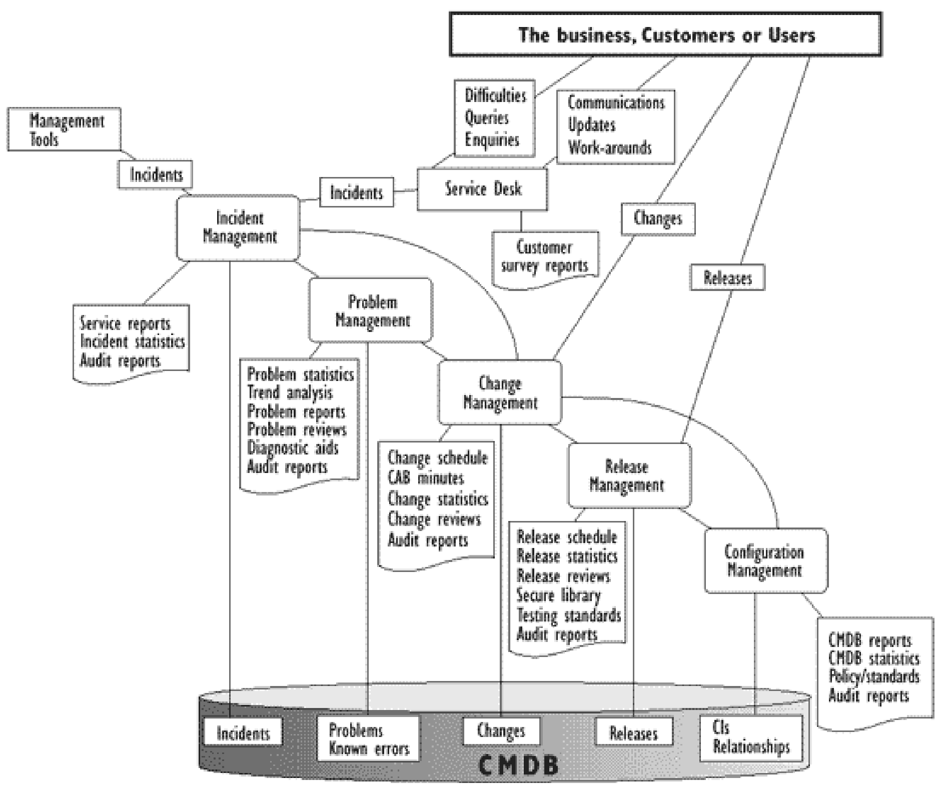

Figure 20. ITIL conceptual model (Berkhout, Harrow et al. 2000)

This clearly has substantial overlaps with the concept of metadata. It is a core thesis of this capstone project that resolving these overlaps and potential contradictions is critical to the evolution of metadata, IT service management, and enterprise IT generally.

The authors of ITIL also seem to recognize the scope of their problem, and (in 1996) stated that

> …there is no known product that addresses the required functionality of all five service delivery disciplines within a single tool, let along one that also integrates support for service support. The majority of service delivery products currently available address a single discipline … **The question of interfaces between tools from different suppliers is therefore likely to be a significant issue.** (CCTA 1996), emphasis added**.**

The final sentence is highlighted, as it is precisely in this area of interfaces that the other major topic of this research, the current state of industry metadata standards, comes into play.

10.2.3. The concept of Configuration Item

Perhaps the most relevant (with respect to traditional metadata) area in the ITIL standards (and in the IT Service Management literature generally) is the concept of Configuration Management (abbreviated CM).

> Configuration Management addresses the control of a changing IT infrastructure (standardization and status monitoring), identifying the configuration items (inventory, mutual links, verification, and registration), collecting and managing documentation about the IT infrastructure **and providing information about the IT infrastructure to all other processes**. (Van Bon, Kemmerling et al. 2002), emphasis added.

Or, more succinctly, "Items that should be under the control of Configuration Management include hardware, software and associated documentation…"

Contrast the assertion that CM is responsible for "providing information about the IT infrastructure to all other processes," with this previously cited (Marco 2003) quote that the metadata repository is the:

> . . . system that manages a company's systems. A meta data repository catalogs all of the applications, data, processes, hardware, software (technical meta data), and business knowledge (business meta data) possessed by an organization.

The key difference is that CM process explicitly includes data entry, unlike the "automated" metadata vision presented above. Configuration Management includes the responsibility of:

> "…Selecting and identifying the configuration structures for all the infrastructure's **CI**s, including their 'owner', their interrelationships and configuration documentation. It includes allocating identifiers and version numbers for **CI**s, labelling each item, and entering it on the **Configuration Management Database** (**CMDB**)." (Berkhout, Harrow et al. 2000), section 7.2.

I argue that the metadata repository can **never** be considered a "system that manages a company's systems" as long as it has no initial data entry/capture function. The metadata repository would need to be a system of record in order to lay claim to such a management function. The CMDB is such a system; the metadata repository – as long as it remains passive – can never be.

Possible Configuration Item types to be stored in the CMDB include (Berkhout, Harrow et al. 2000) section 7.6.2:

- hardware (including network components where relevant)
- system software, including operating systems
- business systems - custom-built applications
- software packages
- database products
- physical databases
- environments
- feeds between databases, applications and EDI links
- configuration baselines
- software releases
- configuration documentation, e.g. system and interface specifications, licences, maintenance agreements, SLAs, decommissioning statement
- Change documentation, deviations and waivers
- other resources e.g. Users, suppliers, contracts
- other documentation e.g. IT business processes, workflow, procedures
- network components
- Service Management components and records such as capacity plans, IT service continuity plans, Incidents, Problems, Known Errors, RFCs, etc.

The overlap with metadata as defined in section 4. is notable. And while (Berkhout, Harrow et al. 2000) stipulate (7.5.6) that the CMDB team is to "review configuration data held in hard-copy form, in local spreadsheets or in databases, and develop a conversion/loading strategy" they do not use or seem to recognize the concept of metadata. It is my assertion that **the two disciplines are clearly on collision course,** an observation I believe is original.[20]

10.2.4. Configuration Item definition: another term for metamodeling?

The concept of a Configuration Item (CI) is key to the CMDB – everything within the CMDB is a CI. The Configuration Management business processes are all based around the concept of CIs, a concept that predates ITIL and ITSM (Sisley 2003).

Consider the following extended passage from (Berkhout, Harrow et al. 2000):

> "Configuration structures should describe the relationship and position of CIs in each structure… CIs should be selected by applying a decomposition process to the top-level item using guidance criteria for the selection of CIs. A CI can exist as part of any number of different CIs or CI sets at the same time… The CI level chosen depends on the business and service requirements.
>
> Although a 'child' CI should be 'owned' by one 'parent' CI, it can be 'used by' any number of other CIs…
>
> Components should be classified into CI types…Typical CI types are: software products, business systems, system software.…The life-cycle states for each CI type should also be defined; e.g. an application Release may be registered, accepted, installed, or withdrawn…
>
> The relationships between CIs should be stored so as to provide dependency information. For example, … a CI is a part of another CI[,] … a CI is connected to another CI [,] … a CI uses another CI…

A data analyst reading the above passage as a requirements specification would probably start to build an information model looking something like this:

[20] Also supported by numerous conversations I have had in the past 2 months with leading practitioners in both fields, who are just now waking up to this.

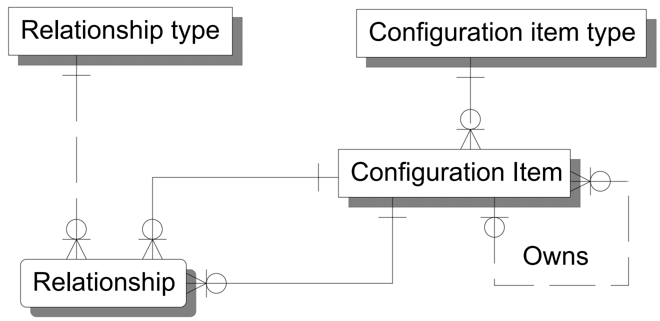

Figure 21. Basic configuration item metamodel

Note the similarity to Figure 11. This is a highly flexible and abstract data structure; the core of it (Configuration Item and Relationship) essentially is a data model that can contain graphs. (As noted above, (Carlis and Maguire 2001) cover data structures required for graphs (as well as many other standard computer science data structures) in some detail (p. 347-349).)

Generally, use of such flexible data structures can be problematic. The question of the relationship concept alone is so fundamental to computer science and information system design that the ISO has devoted a whole publication to it, the "General Relationship Model." (International Standards Organization 1995) It is not clear that the ITIL authors have incorporated this level of information theory into their conceptions, and this lack of precision in defining relationships between CIs may have unfortunate ramifications for the reliability and interoperability of CMDB-stored information.

ITIL seems to encourage the configuration management team to define and relate CIs with few constraints: the CM team is to "set up CI types, attributes, types of relationships, high-level CIs." Pitfalls are identified:

> "Possible problems faced in **Configuration Management** are: **CI**s are defined at the wrong level with too much detail (so that staff become involved in unnecessary work) or too little detail (so that there is inadequate control)." (Berkhout, Harrow et al. 2000) section 7.4.2

I assert that **the definition of configuration items in a Service Management Framework configuration management tool *is* in a sense metamodeling.** Anyone attentive to the deep debates around the Object Management Group's standard models and metamodels knows that metamodeling is a non-trivial, systems-level analysis and design task. Putting it into the domain of end-user configuration (even when those end users are IT administrators) may be problematic.

Using a tool based on this specification,[21] an administrator conceivably could create an application system, a RAM chip, a database table, and relate them all arbitrarily without regard to what semantically makes sense. (For example, a RAM chip should not be linked directly to a database table, but only via the server to which it is providing the service of main memory management.)

The issue is well recognized in OMG metamodeling:

> "How can classes be kept from owning other classes that they are not allowed to own? For example, what prevents an XML Schema from owning a Relational Trigger?…The CWM specification contains rules, *integrity constraints,* which are encoded in a special purpose language, OCL [Object Constraint Language]." (Poole, Chang et al. 2002) p. 114.

The general theory of configuration item definition, at least as presented in ITIL, apparently lacks the ability to describe such constraints. Vendors may implement some functionality along these lines, but since there is no normative metamodel specified, such functionality would be proprietary.

The more robust OMG and DMTF models appear at risk of being ignored by modern configuration management (and ITSM more generally). It is true that these models are very intricate, and yet their ability to handle the entire web of complex dependencies in the IT management space has yet to be proven. (This will be covered in greater depth in following sections.) However, there is no question that some risk exists of significant reinvention of metamodel wheels under the auspices of ITSM and the CMDB in enterprises worldwide.[22]

[21] No actual CMDB tools were reviewed for this research. It is possible that some vendors have recognized this as a potential issue. However, the main point still stands: this **is** metamodeling.

[22] It may be surprising, but this author believes this paper to be the first in-depth identification and analysis of this problem.

10.2.5. Configuration management architecture

Finally, Configuration Management has another significant overlap with historical metadata in that the ITIL standard calls for automated scanning that appears essentially identical to the architectures used by metadata repositories (see "Metadata architecture," p. 18).

> Automated processes to load and update the Configuration Management database should be developed where possible so as to reduce errors and reduce costs. Discovery tools, inventory and audit tools, enterprise systems and network management tools can be interfaced to the CMDB. These tools can be used initially to populate the CMDB, and subsequently to compare the actual 'live' configuration with the records stored in the CMDB. (Berkhout, Harrow et al. 2000) section 7.3.7.

One **major** difference between metadata tools and the configuration management space is that configuration management explicitly calls for integration with management frameworks (*e.g.,* HP Openview, CA Unicenter, or IBM Tivoli) (Berkhout, Harrow et al. 2000) section 7.9.1; metadata repositories have not historically sought integration into operational systems of this type[23].

On the other hand, ITIL recognizes (Berkhout, Harrow et al. 2000), 7.9.5 that "There are many support tools that can assist Change Management, Configuration Management and Release Management." Examples cited **(Berkhout, Harrow et al. 2000)** include:

- **requirements analysis and design tools, systems architecture and CASE tools**
- **database management audit tools**
- document-management systems
- distribution and installation tools
- comparison tools
- build and release tools
- installation and de-installation tools
- compression tools
- listing and configuration baseline tools
- audit tools (also called 'discovery' or 'inventory' tools)
- detection and recovery tools
- reporting tools

The items represented in **boldface** are also historically used as sources to metadata repositories. But metadata repositories and technology are mentioned nowhere in the ITIL spec. Generally, the different sources demonstrate the orientation of metadata to the systems

[23] This assertion is based on the author's extensive reviews of repository products in the course of career activities, including the Platinum, ASG-Rochade, Adaptive, Data Advantage Group, and MetaMatrix products.

development lifecycle, and configuration management to IT operations. Nevertheless, the contention of this research stands: the two domains are converging.

10.3. Limitations of process-centricity

ITIL is primarily described in terms of a process structure (actually, more of a set of prescriptive requirements for what good process should look like, but it is not difficult at all to infer a reference process from these requirements). The immediate question that comes to the mind of the software engineer is "what about the data?" Processes require management of persistent state, and best practices for organizing that persistent state have been evolving for decades. While the process-centric view of the world focuses on decomposition, steps, inputs & outputs, and control, the data-centric view is never far away – the focus on measurable results ensures this; without data there is nothing to measure.

The concepts of a 3rd normal form logical data model or a data flow diagram are well-established partners to process modeling in the definition of systems and solutions. The first-generation ITIL specifications did include data flow diagrams (CCTA 1996), but more recent ITIL work has been purely prescriptive, with little normative aspect. This appears to be a step backwards.

With this piece of its foundation missing, ITIL has been challenged:

> "The ITIL puzzle has also been compared with tectonic plates, or colliding and overlapping continents. Not only is it difficult to identify the boundaries exactly, there is also clearly friction and stress . . . The complex interrelationship between the processes described in the books on Service Support and Service Delivery is almost impossible to show in a diagram." (Van Bon, Kemmerling et al. 2002) p. 34-35.

It would be interesting to apply full parallel decomposition, CRUD matrixing, and affinity analysis to an ITIL reference model to more precisely determine the nature of these so-called "tectonic interfaces." It seems like that these issues might be closely related to issues that also make metadata a complex and intractable subject. As (Cassidy and Guggenberger 2001) observe,

Interestingly, the Troux enterprise architecture product **does** have management framework integration, but no integration with modeling tools.

> In many areas of the business, processes are repetitive and produce a tangible or deliverable product. Information Systems . . . is more of a job-shop environment, as no two project or problems are really alike. Processes are more difficult to identify, document, and automate in a variable environment. **However, when complex problems are broken down into simple steps, commonality can be found**.
>
> Information Systems is also a relatively newer discipline than manufacturing and other service areas. There is less structure and organization, and fewer standards have been developed in the information systems industry as a whole compared with other industries (emphasis added).

It is a core thesis of this work that when these complex problems are broken down into their "simple steps," at least the data elements of these steps can mostly be found in the OMG and DMTF metamodels. Mostly – the standards bodies have not yet succeeded in a comprehensive, integrated vision of strongly and weakly typed metadata, which will be covered in the next major section.

10.4. Configuration management conclusions

This detailed review of CMDB theory and practice leads to my asserting these conclusions:

1. The concepts of the metadata repository and the configuration management database (CMDB) are converging in scope. The differences in emphasis and history between these two conceptions will soon be irrelevant.

2. The current conception of the CMDB from the Information Technology Infrastructure Library (ITIL) and other IT Service Management sources pays insufficient attention to the data structures required to meet CMDB requirements. These data structures– including the user-defined Configuration Item concept– are metamodels, and the data they contain is metadata (data about data and the systems that manage it).

3. Metamodeling is a complex and risky endeavor, with substantial standards activity. CMDB vendors, ITSM theorists, and the ITIL standard should start to explicitly look to the OMG and DMTF standards for reference definitions of configuration items.

4. Tools currently sold as CMDB solutions may lock enterprises into proprietary vendor metamodels, resulting in minimal interoperability at a time when standards are available from the DMTF and OMG to mitigate such lock-in.

5. No single vendor can meet all of the requirements for comprehensive, end-to-end enterprise IT metadata, given the wide variety of tools and complex, specialized problem domains in the IT service function. Industry standards for IT metadata interchange are essential to the success of the configuration management (and broader IT Service Management) vision. This lesson was learned some time ago by CASE and metadata vendors, who have been supporting the OMG's standards efforts. The new generation of ITSMF vendors, however, appears unaware of these standards.[24]

6. A tremendous opportunity presents itself in the potential alignment of ITIL process rigor with OMG metamodel rigor. Each has something the other needs. The emergence of Model-Driven Architecture in the software development lifecycle only furthers this need for standards-based alignment.

11. Strong and weak typing in metadata

As we have seen, the metadata problem in enterprise IT includes both the challenge of describing instances of very specific objects (tables, columns, servers, components, message queues, and so forth) as well as tracking their arbitrary relationships. In some ways, I believe this problem begins to separate itself into two domains: strongly and weakly typed. Both are needed for effective enterprise metadata management.

11.1. Strong typing

Strongly typed metadata can be seen in the classic data dictionary. This is a very familiar approach to any data modeler:

- explicitly define the entity, its attributes, and their data types.
- define relationships (identifying or non-identifying) between the entities.

[24] This is based on the author's career experience, recently including evaluations of products from ITSM-branded vendors Axios, Relicore, and HP. Enterprise architecture products such as Envision, P-Tech, and Troux also do not currently support industry standards.

This works well for metadata domains such as the UML or the CWM. However, the general information supply chain is more problematic: if it is not abstracted into the graph problem domain, but left in the domain of specific types and their relationships, the metadata management system can easily get bogged down in formidable data structures like this:

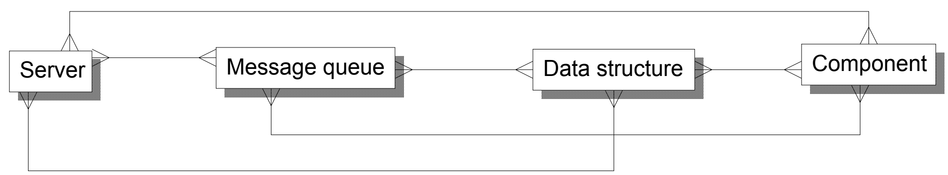

Figure 22. Strongly typed metadata and relationships

Note that the link between each entity is many-to-many. If resolved for implementation using an RDBMS, the number of entities would expand from four to ten, and the SQL used to extract information would be quite complex, requiring recursion and much procedural scaffolding to traverse the paths.

Imagine the information supply chain problem represented in this strongly typed world. These structures could describe a set of relationships that is essentially a graph. However, as illustrated in Figure 9 and Figure 10, **the immediate pragmatic problem is that there is no easy way to separate out distinct paths from the (possibly massive) graph**, which is necessary if the metadata is to be used effectively for supporting particular interaction or integration scenarios. Paths are necessary to make such a graph human-understandable, which brings us into the realm of weak typing.

One major disappointment with the OMG standards is that their semantics for handling the above-described integration metadata problem (OMG 2002f; OMG 2002b) are highly complex, bordering on unusable. This is because they are quite strongly typed.

11.2. Weak typing

By weakly typed I mean that every object in a given metadata management domain has the same type. It is metadata that uses as a primary management strategy objects of the same data type to represent very diverse real world subjects. This is necessary for example for participation in graphs and related data structures.

As we have seen, there are emergent areas in metadata management that do not lend themselves well to a strongly typed approach:

Configuration Management. IT Service Management internal business processes around change and configuration management call for a common concept of a configuration item (Berkhout, Harrow et al. 2000; Van Bon, Kemmerling et al. 2002) that might be applied as a weakly typed overlay to many of the metadata types. See Appendix I: Reference Configuration Items.

Search indexing. Requirements for an integrated search capability across the metadata repository also drive a weakly typed approach. The award-winning metadata management practice at Bell South, for example, extensively uses the Dublin Core standard, which is based on fifteen common attributes used across entire repositories for searching (Stephens 2003).

Integration. Per the above discussion on the problems of integration metadata, it also can be seen as weakly typed, to the extent that all objects share a Node identity that enables their participation in graphs and paths.

Reuse. The Rational Reusable Asset Specification (Rational Software Corporation 2003) is another clear example of a weakly typed metamodel; the core entity is simply called Asset, surrounded by relatively few supporting classes.

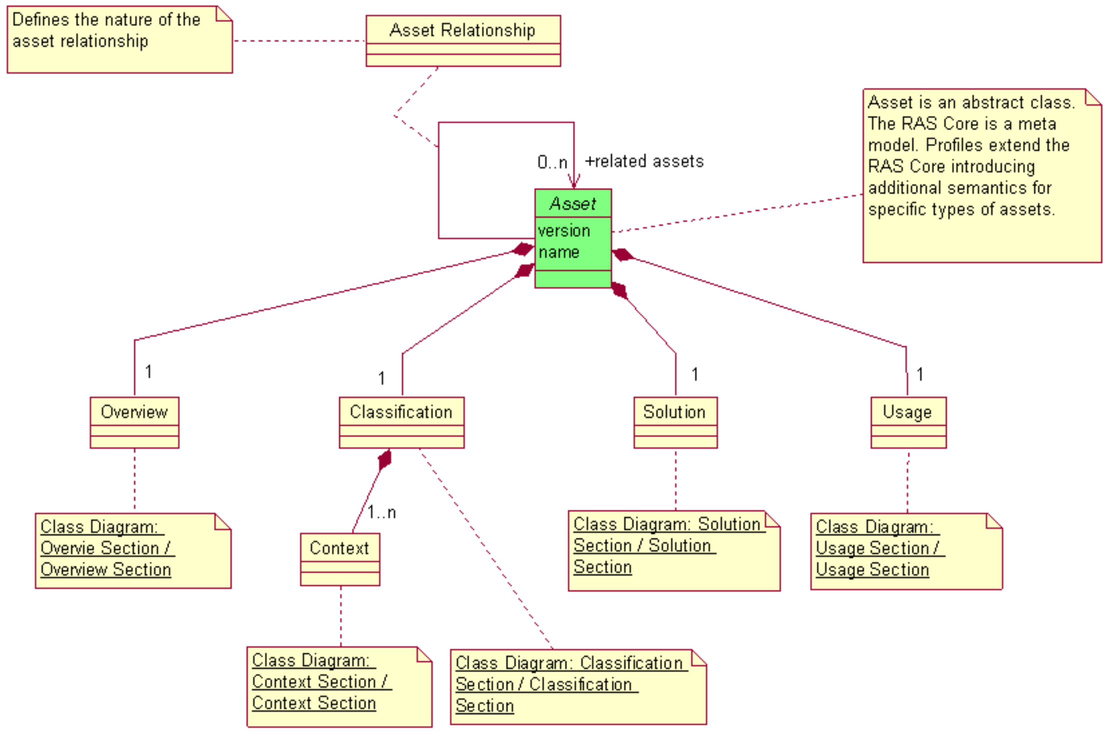

Figure 23. Rational Reusable Asset Specification

As with strong typing, weak typing also can lead to problems. Consider the following ITIL passage:

> Some organisations devolve control to support groups that have expertise in a particular technology or platform, because it may not be cost-effective to train central staff in specialist areas. In these cases, the support group manager is responsible for the control of **CI**s owned and maintained by the group. The organisation's procedures for Change Management, Configuration Management, Release Management and a centralised **CMDB** should be adopted wherever possible. (Berkhout, Harrow et al. 2000) section 7.5.5

While it is not clear exactly what the ITIL authors meant, this researcher assumes that examples of such areas would be:

- data and database administration
- messaging administration
- ETL (extract, transform, load) administration

These areas use highly specialized tools and consoles that will not – out of the box – interoperate with proprietary CMDB tools. The concept of CI is usually alien to such tooling; the metamodels are generally strongly typed. What is necessary is a means of integrating strongly and weakly typed metadata.

11.3. Harmony between strong and weak

Let's assume that the Node entity is seen as a supertype for a number of distinct types. In object-oriented terms, each of these subtypes could be said to implement a Node interface as well as its specific characteristics. Such a structure would look like this:

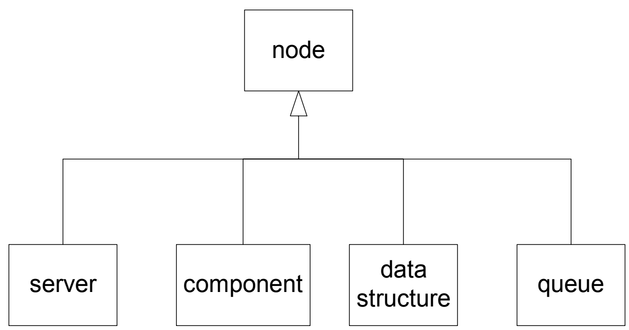

Figure 24. Integrating strongly typed metadata into a graph context

This solution presents some very interesting problems, however. There are legitimate relationships among strong types (strict composition for example) that would be ill-served by migrating up to the generalized graph level. There are also the above-cited problems in determining relationship types, and constraining the creation of semantically invalid linkages.

How would one partition the two domains? One possible solution I propose would be to implement an extended graph metamodel that also defined semantically valid linkages via some declarative constraint mechanism. Here for example is an object-oriented rendition of the graph-centric, weakly typed data model in Figure 21, with some extensions to enforce semantic validity:

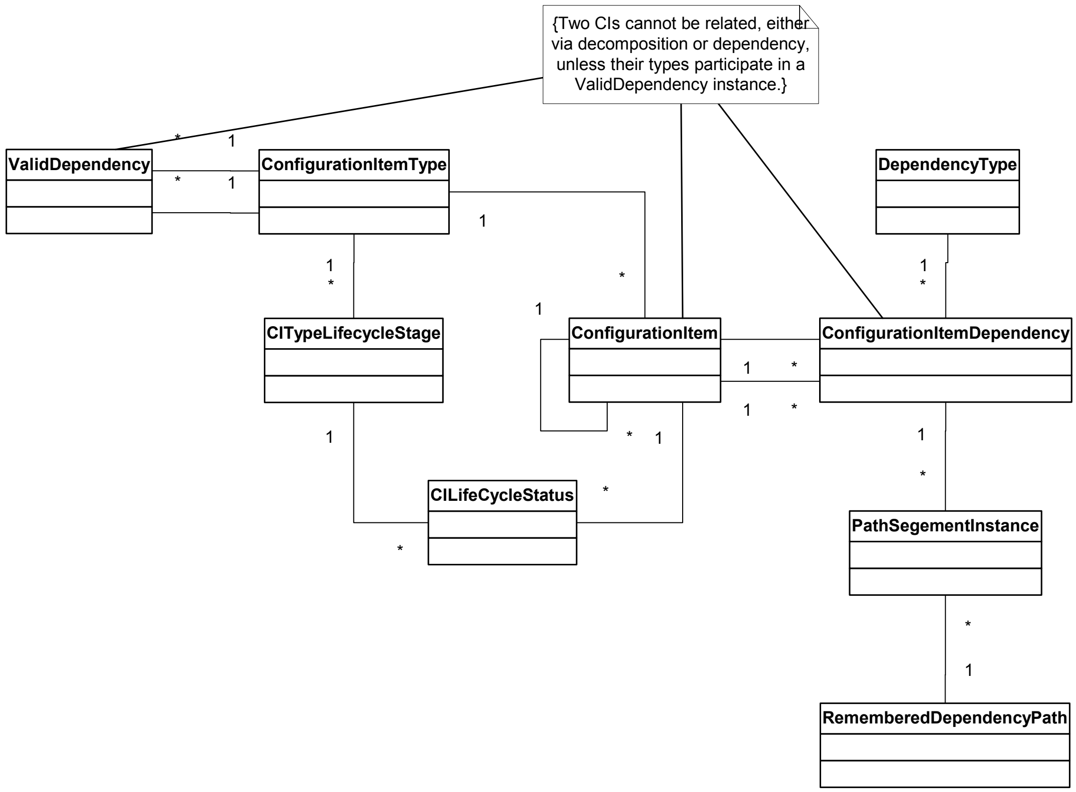

Figure 25. Elaborated configuration item metamodel

I believe the establishment of such a standard for a configuration management metamodel by either the OMG or the DMTF, in partnership with the ITSMF, would seem to be a key step in rationalizing and aligning the process-centric and information-centric standards.

12. Relating the standards

A major question facing the IT standards bodies is whether and to what extent they will seek to align with each other, rationalizing areas of overlap and promoting some level of orthogonality between process and information. This would enable the matrixing of the process-centric and information-centric standards.

Matrixing function or process to data is of course a long-standing analysis technique in systems development ((Martin 1989b), p. 171). For example, (Tannenbaum 2002) on page

287 presents a simple CRUD matrix[25] that cross-references the major data staff roles with the major metadata subjects; here is a slight adaptation:

	Data administration	Data design	Database administration
Subject Area	C	R	
Data model		C	R
Physical schema			C

Table 4. Matrix of data management process and metadata domains

This example is a subset of what might result from a comprehensive cross-referencing of the ITIL and CMM process models with the OMG and DMTF information metamodels.

The following overall process/data matrix framework is proposed. The CMM and ITIL standards are seen as process-centric, while the OMG and DMTF standards are seen as data-centric. The OMG and DMTF in general have not defined prescriptive standards or process models such as those covered by ITIL or the CMM.[26]

	Process		
Data		CMM	ITIL
	OMG metamodels		
	DMTF metamodels		

Table 5. Matrix of process-centric and information-centric standards

Such an effort, while outside the scope of this paper, would be a major step forward in aligning the work of the standards bodies.

[25] C, R, U, D: Create, Read, Update, Delete. This technique for matrixing function or process to data is of course a long-standing analysis technique in systems development Martin, J. (1989b). *Information Engineering Book II: Planning & Analysis*. Englewood Cliffs, N.J., Prentice Hall., p. 171. Object orientation has moved away from this technique, but for a top-down analysis given the current set of initial conditions, this author considers it ideal.

[26] However, the OMG is working on **languages** for process definition and business rules management. If reference IT management processes were defined using a MOF-based process modeling language, the synergies could be powerful.

13. Conclusion

In this paper, I have argued for and I believe supported the following conclusions:

Radical transparency. The problems of enterprise IT are fundamentally due to a lack of holistic visibility and traceability from the highest-level business requirements to the lowest level technology. Of the major enterprise resource areas, only information technology itself has no integrated systems dedicated to providing comprehensive, structured information management services. While the problems and risks of enterprise resource planning are well known (Glass 1998; CIO Magazine 2002), the alternative (small, silo applications with very poor data integrity) is also not working for enterprise IS.

Convergence is happening. Various approaches and techniques have attempted to solve parts of the IT crisis. Metadata is perhaps the pre-eminent or foundational discipline, but suffers from a number of inherent difficulties in implementation. Beyond metadata, no one vendor or standard has claimed hegemony, but there is a clear convergence of interest in and motion towards the common "ERP for IT" space.

Process-centric standards need to be aligned with normative information models. The prescriptive process standards represented by the CMM and SEI usually invoke this reaction: "They tell you what to do, but not how." On the other hand, the OMG and DMTF are clearly outlining the **how,** at least in the crucial information model space. The two general approaches are clearly complementary, yet no efforts have been made to reconcile them.

Both strong and weak typing are needed for the IT problem domain. The world of large-scale enterprise IT cannot be effectively represented with exclusively a strongly typed or weakly typed approach in metamodeling. An integrated solution using both is needed.

Standards-based ERP for IT would be preferable to vendor domination. The absence of the proverbial 800-pound gorilla ERP vendor may prove an asset, if it results in the creation of an ecosystem of vendors focused on providing an interoperable, standards-based solution for the problems of 21st century information technology. A number of vendors are emerging in this area; however, they seem oblivious (or even resistant) to important industry standards that could make their products much more interoperable.

The large IT organizations must take the initiative in insisting on a standards-based approach. IT is a highly complex domain to successfully model and no one vendor can do it all. The activation and operational success of the OMG and DMTF standards, and the explicit reconciliation of ITSM processes with the OMG or DMTF, would move technical innovation another layer higher in the semantic stack, to the benefit of all.

14. Appendix I: Reference Configuration Items

From Annex 7C: Suggested CI attributes (Berkhout, Harrow et al. 2000)

Attribute	Description
CI Name	The unique name by which this type of CI is known.
Copy or Serial Number	The number that uniquely identifies the particular instances of this CI - for example, for software the copy number, for hardware the serial number.
Category	Classification of a CI (e.g. hardware, software, documentation etc).
Type	Description of CI type, amplifying 'category' information (e.g. hardware configuration, software package, hardware device or program module).
Model Number (hardware)	Model of CI (corresponding, for example, to supplier's model number e.g. Dell model xxx, PC/aa model yyy).
Warranty expiry date	Date when the supplier's warranty expires for the CI.
Version Number	The version number of the CI.
Location	The location of the CI, e.g. the library or media where the software CIs reside, the site/room where a service is located.
Owner Responsible	The name and/or designation of the owner responsible for the CI.
Responsibility Date	Date the above owner became responsible for the CI.
Source/supplier	The source of the CI, e.g. developed in-house, bought in from company xxxxx etc.
Licence	Licence number or reference to licence agreement.
Supply Date	Date when the CI was supplied to the organisation.
Accepted Date	Date when the CI was accepted by the organisation as satisfactorily tested.
Status (current)	The current status of the CI; e.g. under 'test', 'live', 'archived'.
Status (scheduled)	The next scheduled status of the CI (with the date or indication of the event that will trigger the status change).
Parent CI(s) relationships	The unique CI identifier(s) - name/copy/number/model/number/ of the 'parent(s)' of this CI.
Child CI(s) relationships	The unique CI identifier(s) of all 'children' of this CI.
Relationships	The relationship of the CI with all CIs other than 'parent' and 'child' (e.g. this CI 'uses' another CI, this CI 'is connected to' another CI, this CI is 'resident on' another CI, this CI 'can access' another CI).
RFC Numbers	The identification number of all RFCs affecting this CI.
Change Numbers	The identification number of all Change records affecting this CI.
Problem Numbers	The identification number of all Problem records affecting this CI.
Incident Numbers	The identification number of all Incident records affecting this CI.
Comment	A comment field to be used for textual narrative; for example, to provide a description of how this version of the CI is different from the previous version.

15. References

Alanen, M. (2002). "A Meta Object Facility-Based Model Repository with Version Capabilities, Optimistic Locking and Conflict Resolution". *Computer Engineering*, Abo Akademi University**:** 85.

Allegra, P. (2003). "Modeling the Enterprise Solution Architecture", META Group. **2003**.

Atkinson, C., B. Henderson-Sellers, et al. (2000). "To Meta or Not to Meta -- That Is the Question." *Journal of Object-Oriented Programming*.

Bach, J. (1998). "Microdynamics of Process Evolution." *IEEE Computer* **31**(2): 111-113.

Bamberger, J. (1997). "Essence of the Capability Maturity Model." *IEEE Computer*: 112-114.

Berkhout, M., R. Harrow, et al. (2000). "Service Support: Service Desk and the Process of Incident Management, Problem Management, Configuration Management, Change Management and Release Management". London, The Stationery Office.

Bezivin, J. (1998). "Who's Afraid of Ontologies?". 13th Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA 1998), Vancouver, BC.

Bézivin, J. (2001). "From Object Composition to Model Transformation with the MDA". **2002**.

Bezivin, J. and S. Gerard (2002). "A Preliminary Identification of MDA Components". 17th Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA 2002), Seattle, Washington.

Bezivin, J. and N. Ploquin (2001). "Tooling the MDA framework: a new software maintenance and evolution scheme proposal." *Journal of Object-Oriented Programming*.

Boehm, B. (1988). "A Spiral Model of Software Development and Enhancement." *IEEE Computer* **21**(5): 61-72.

Boehm, B. (2000a). "Gaining Intellectual Control of Software Development." *IEEE Computer* **33**(5): 7-33.

Boehm, B. (2000b). "Unifying Software Engineering and Systems Engineering." *IEEE Computer*.

Boehm, B., D. Port, et al. (2000). "Avoiding the Software Model-Clash Spiderweb." *IEEE Computer*: 120-122.

Brackett, M. H. (2000). *Data resource quality : turning bad habits into good practices*. Boston, Addison-Wesley.

Bumpus, W. (2000). *Common information model : implementing the object model for enterprise management*. New York, N.Y, J. Wiley.

Butler, C. W. and J. D. Clark (1999). "Breaking the Data Ties that Bind You". *IT Professional (IEEE)***:** 46-50.

Carlis, J. V. and J. D. Maguire (2001). *Mastering data modeling : a user-driven approach*. Boston, Addison-Wesley.

Cassidy, A. and K. Guggenberger (2001). *A Practical Guide to Information Systems Process Improvement*. Boca Raton, FL, St. Lucie.

CCTA (1996). *IT Service Delivery Tools*. London, HMSO.

Celko, J. (2000). *Joe Celko's SQL for smarties : advanced SQL programming*. San Francisco, Morgan Kaufmann.

CIO Magazine (2002). "Enterprise Resource Planning Overview". *CIO Magazine Online*.

Cook, M. A. (1996). *Building enterprise information architectures : reengineering information systems*. Upper Saddle River, NJ, Prentice Hall.

Crawley, S., S. Davis, et al. (1997). "Meta-meta is better-better!". International Working Conference on Distributed Applications and Interoperable Systems (DAIS '97).

Dama International (2000). *Implementing Data Resource Management*, DAMA International.

De Lucia, A., A. Pannella, et al. (2001). "Assessing Massive Maintenance Processes: An Empirical Study". IEEE International Conference on Software Maintenance (ICSM'01), Florence, Italy.

DeMarco, T. (1979). *Structured analysis and system specification*. Englewood Cliffs, NJ, Yourdon Press.

Demeyer, S., S. Ducasse, et al. (1999). "Why Unified is not Universal: UML Shortcomings for Coping with Round-trip Engineering". The Second International Conference on The Unified Modeling Language, Fort Collins, Colorado, USA, Springer.

Distributed Management Task Force (2000). "Common Information Model (CIM) Core Model".

Distributed Management Task Force (2002a). "CIM Database Model White Paper". **2003**.

Distributed Management Task Force (2002b). "Specification for the Representation of CIM in XML".

Distributed Management Task Force (2003). "Common Information Model Schema". **2003**.

Distributed Management Task Force, T. F. (2002c). "TeleManagement Forum Liaison Activities – Work Register".

Ernst, J. (1997). "Introduction to CDIF". **2003**.

Eshuis, R. and R. Wieringa (2001). "A Formal Semantics for UML Activity Diagrams--Formalising Workflow Models". **2002**.

Finkelstein, C. (1989). *An introduction to information engineering : from strategic planning to information systems*. Sydney, Australia ; Reading, Mass., Addison-Wesley Pub. Co.

Flatscher, R. G. (1996). "An Overview of the Architecture of EIA's CASE Data Interchange Format (CDIF)." *Informationssystem Architekturen* **3**(1): 26-30.

Fowler, M. and K. Scott (2000). *UML distilled : a brief guide to the standard object modeling language*. Reading, Mass., Addison-Wesley.

Frankel, D. S. (2003). *Model-Driven Architecture: Applying MDA to Enterprise Computing*. New York, John Wiley & Sons.

Glass, R. L. (1998). *Software runaways*. Upper Saddle River, NJ, Prentice Hall PTR.

Goldfine, A. and P. Konig (1988). "A Technical Overview of the Information Resource Dictionary System". Gaithersburg, MD, National Institute of Standards and Technology.

Gray, L. (1998). "Gray Rebuts Bach: No Cowboy Programmers!" *IEEE Computer* **31**(4): 102-103.

Grose, T. J., G. C. Doney, et al. (2002). *Mastering XMI : Java programming with XMI, XML, and UML*. New York, John Wiley & Sons.

Hartman, F. A., Rafi A. (2002). "Project Management in the Information Systems and Information Technologies Industries." *Project Management Journal* **33**(3): 5-15.

Hayes, F. (2002). "Practice Questions". *Computerworld*: 58.

Holmes, N. (2000). "Why Johnny Can't Program." *IEEE Computer* **33**(12): 158-159.

Huang, K.-T., Y. W. Lee, et al. (1999). *Quality information and knowledge*. Upper Saddle River, N.J., Prentice Hall PTR.

Huc, C., T. Levoir, et al. (1997). "Metadata: models and conceptual limits". The Second IEEE Metadata Conference, Silver Spring, Maryland, USA.

Hughes, E., A. Rosenthal, et al. (1999). "Toward A Methodology for Enterprise Components".

Humphrey, W. S. (1989). *Managing the software process*. Reading, Mass., Addison-Wesley.

Humphrey, W. S. (1994). "The Personal Process in Software Engineering". ICSP 3 - 3rd International Conference on Software Process, Reston. USA.

Inmon, W. H. (1999). *Building the Operational Data Store*. New York, John Wiley & Sons.

Inmon, W. H. (2002). "Toward a Unified Theory of Meta-Data." *The Data Administration Newsletter*(22).

International Standards Organization (1995). "Information Technology - Open Systems Interconnection - Management Information Services - Structure of Management Information - Part 7: General Relationship Model".

IT Service Management Forum (2003). "IT Service Managment Forum USA 2003 Attendee FAQ". **2003**.

Jacobson, I., G. Booch, et al. (1999). *The unified software development process*. Reading, Mass, Addison-Wesley.

Jahn, K. (2003). "Where's Your Backbone? Effective enterprise data delivery requires more than just a traditional data warehouse". *Intelligent Enterprise*. **6:** 33-36.

Kan, S. H. (1995). *Metrics and models in software quality engineering*. Reading, Mass., Addison-Wesley.

Kan, S. H., V. R. Basili, et al. (1994). "Software quality: An overview from the perspective of total quality managment." *IBM Systems Journal* **33**(1): 4-19.

Kan, S. H., S. D. Dull, et al. (1994). "AS/400 software quality management." *IBM Systems Journal* **33**(1): 62-88.

Keller, R. K., J.-F. Bedard, et al. (2001). "Design and Implementation of a UML-based Design Repository". Proceedings of the Thirteenth Conference on Advanced Information Systems Engineering (CAiSE'01),, Interlaken, Switzerland.

Kimball, R. and M. Ross (2002). *The data warehouse toolkit : the complete guide to dimensional modeling*. New York, Wiley.

Kullbach, B., A. Winter, et al. (1998). "Program Comprehension in Multi-Language Systems". Conference on Software Maintenance and Reengineering (WCRE '98).

Larman, C. (2001). "The Business Case for the Iterative Unified Process". **2001**.

Loshin, D. (2001). *Enterprise knowledge management : the data quality approach*. San Diego, Calif.; London, Morgan Kaufmann ; Academic Press.

Marco, D. (2000). "Building and managing the meta data repository : a full lifecycle guide". New York, John Wiley.

Marco, D. (2003). "Meta Data Repository: A System that Manages Our Systems". *The Data Administration Newsletter #23*, Robert Seiner. **2003**.

Martin, J. (1989a). *Information Engineering Book I: Introduction*. Englewood Cliffs, N.J., Prentice Hall.

Martin, J. (1989b). *Information Engineering Book II: Planning & Analysis*. Englewood Cliffs, N.J., Prentice Hall.

Martin, J. (1989c). *Information Engineering Book III: Design & Construction*. Englewood Cliffs, N.J., Prentice Hall.

McClure, C. (1989). *CASE is Software Automation*. Englewood Cliffs NJ, Prentice Hall.

Mylopoulos, J. (2001). "Metamodeling".

Narayan, R. (1988). *Data dictionary : implementation, use, and maintenance*. Englewood Cliffs, N.J., Prentice Hall.

Ogbuji, U. (2002). "XML, The Model Driven Architecture, and RDF". XML Europe 2002, Barcelona, Spain.

Olson, J. (2002). "Data Profiling: The Key to Success in Integration Projects". *EAI Journal*. **4:** 22-26.

OMG (1998). "Software Process Engineering Request for Information", Object Management Group.

OMG (2002a). "Common Warehouse Metamodel (CWM) Specification", Object Management Group.

OMG (2002b). "Enterprise Distributed Object Computing Metamodel", Object Management Group.

OMG (2002c). "Meta Object Facility (MOF) Specification Version 1.4", Object Management Group.

OMG (2002d). "Software Portfolio Management Facility", Object Management Group.

OMG (2002e). "Software Process Engineering Metamodel", Object Management Group.

OMG (2002f). "UML Profile for EAI", Object Management Group.

OMG (2002g). "Unified Modeling Language v. 1.4". Needham, MA, Object Management Group.

Page-Jones, M. (1988). *The practical guide to structured systems design*. Englewood Cliffs, N.J., Prentice Hall.

Paulk, M. C. (1995). *The capability maturity model : guidelines for improving the software process*. Reading, Mass., Addison-Wesley Pub. Co.

Perrochon, L. and W. Mann (1999). "Inferred Designs". *IEEE Software*. **16:** 46-51.

Poole, J., D. Chang, et al. (2002). *Common Warehouse Metadmodel: An Introduction to the Standard for Data Warehouse Integration*. New York, Wiley Publishing.

Poole, J., D. Chang, et al. (2003). *Common Warehouse Metamodel: Developer's Guide*. New York, John Wiley & Sons.

Rajlich, V. T. and K. H. Bennett (2000). "A Staged Model for the Software Life Cycle." *IEEE Computer* **33**(7): 66-71.

Rational Software Corporation (1998). "Rational Unified Process: Best Practices for Software Development Teams".

Rational Software Corporation (2003). "Reusable Asset Specification". **2003**.

Redman, T. C. (2001). *Data quality : The Field Guide*. Boston, Digital Press.

Rosen, K. H. (1995). *Discrete mathematics and its applications*. New York, McGraw-Hill.

Rosenthal, A. (1998). "THE FUTURE OF CLASSIC DATA ADMINISTRATION: Objects + Databases + CASE". **2002**.

Rossiter, B. N., D. Nelson, et al. (2001). "A Universal Technique for Relating Heterogeneous Data Models". International Conference on Enterprise Information Systems, Setubal, Portugal.

Schmidt, J. (2003). "From Y2K to P2P The Next Big Embarrassment for the IT Industry?". *EAI Journal*. **4:** 30-33.

Schulte, R., M. Blechar, et al. (2002). "Strategies for Managing Application Integration Metadata", Gartner Group.

SEI (2003). "Management Practices". **2003**.

Sisley, E. M. (2003). "Personal communication". C. Betz.

Spewak, S. H. and S. C. Hill (1993). *Enterprise architecture planning : developing a blueprint for data, applications, and technology*. Boston, QED Pub. Group.

Spurr, K. and P. Layzell, Eds. (1990). *CASE on Trial*. Chichester, England, John Wiley & Sons.

Stephens, T. (2003). "Enterprise Metadata Implementation: Learning from "Best Practices"". DAMA Internation Symposium and Wilshire Metadata Conference, Orlando, Wilshire Conferences.

Stevens, P. (2002). "Small-scale XMI programming: a revolution in UML tool use?" *Journal of Automated Software Engineering*.

Stone, J. A. (1993). *Inside ADW and IEF: the promise and reality of CASE*. New York, McGraw-Hill.

Sturm, R. and W. Bumpus (1999). *Foundations of application management*. New York, J. Wiley.

Sturm, R., L. Erickson-Harris, et al. (2002). *SLM Solutions: A Buyer's Guide*. Boulder, CO, Enterprise Management Associates.

Tannenbaum, A. (1994). *Implementing a corporate repository : the models meet reality*. New York, J. Wiley.

Tannenbaum, A. (2002). *Metadata solutions : using metamodels, repositories, XML, and enterprise portals to generate information on demand*. Boston, Addison-Wesley.

Tele-Management Forum (2002). "Draft report: Mining Information from the DMTF CIM into the TMF SID".

The Open Group (2002). "The Open Group Architectural Framework (TOGAF), Version 8".

Thomas, I. (1989). "PCTE interfaces: Supporting tools in software engineering environments.". *IEEE Software*. **6:** 15-23.

Tibco Inc. (2002). "The Evolution of Managerial Vision and Agility".

Tozer, G. V. (1999). *Metadata management for information control and business success*. Boston, Artech House.

Tsichritzis, D. a. D. and A. C. Klug (1978). "The ANSI/X3/SPARC DBMS Framework Report of the Study Group on Dabatase Management Systems." *Information Systems* **3**(3): 176-191.

(UK), O. o. G. C. (2002). "ITIL FAQ".

Van Bon, J., G. Kemmerling, et al. (2002). *IT service management : an introduction*. [Canada], itSMF-Canada.

Wertz, C. J. (1989). *The data dictionary : concepts and uses*. Wellesley, MA, QED Information Sciences.

Yourdon, E. (1975). *Techniques of program structure and design*. Englewood Cliffs, N.J., Prentice-Hall.

Zahran, S. (1998). *Software process improvement : practical guidelines for business success*. Reading, Mass., Addison-Wesley Pub. Co.