Whitemarsh
Information Systems Corporation

# Comprehensive Metadata Management

Whitemarsh Information Systems Corporation
2008 Althea Lane
Bowie, Maryland 20716
Tele: 301-249-1142
Email: mmgorman@wiscorp.com
Web: www.wiscorp.com

# Table of Contents

# List of Figures

## Executive Summary

> *No one would ever question why a business needs it's finance books. Well, the metadata repository*[1] *is the business's information systems' books. If you cannot run a good business without the former, you cannot run good information systems environment without the latter.*

A significant portion of the time and costs associated with resolving the Year 2000 problem can be directly attributed to a lack of a quality metadata environment within information systems organizations. The fact that one information system organization within an enterprise had virtually no Year 2000 problem while another organization within that same enterprise was running their information systems shop "24x7" to develop and install Y2K solutions was no accident. The former had a long history of metadata management and the later thought metadata was a wasted overhead expense.

In the development of large data processing projects dealing with enterprise-wide, indispensable business functions, documentation of the design requirements and resulting information system specifications is seldom accomplished such that it is timely, accurate, or complete. That is disastrous for the following three reasons:

- Only the momentous facts that are remembered are recorded.
- As systems are specified, the lower-level design details are redundantly developed, often in conflicting manners.
- As system components are maintained, the efforts are crippled because of the undocumented business knowledge that is essential to understanding the component.

Collectively, the entire set of business information system specifications up through requirements and extending into the "data" that defines, structures, and models the activities of the enterprise are metadata. This paper addresses the need for a comprehensive metadata management environment that is woven into the very fabric of database information system specification, implementation, operation, and evolution so as to successfully specify, design, implement, operate and maintain complex information technology components of enterprises. In support of this objective, the paper addresses the following topics:

[1] At Whitemarsh, our metadata repository is an SQL based database application that is called the Metabase. The term, metabase, was crafted from "metadata database" and was first described by Whitemarsh in its books and literature in the early 1980s. The metabase has been created and implemented by Whitemarsh and delivered to its clients since the late 1970s through a variety of database management systems. A demo copy of the current version of the metabase can be downloaded from the Whitemarsh website, www.wiscorp.com.

- What Is Metadata
- Why Is Metadata Important
- Enterprise Architecture Framework
- Metadata Work Products
- `Use Scenarios

## 1.0 What is Metadata?

The quick answer, of course, is that metadata is data about data. However, that's too cute. More formally the string, metadata is divided into *meta* and *data*. Meta in the Oxford Dictionary means, "something of a higher or second-order kind." The word, data, however is not employed within this paper in its strictest sense, that is, a data item like *Birth date = 03/22/1941*, but in more general sense so as to include unstructured data like text and diagrams.

For the purposes of this paper, the scope of metadata is restricted to Information Technology. Consequently, metadata are the materialized artifacts that define the requirements for, the specifications of, design of, or even executing characteristics of an IT system, or component of that system. "System" here is used in a very broad context. Thus, included within the scope of systems are databases, application systems, and their technology environments. Therefore, metadata is all that which is one or more levels of abstraction removed from the actual databases, applications, or their technology environments. In a computing environment, metadata would therefore include:

- Requirements
- Functional descriptions
- Work plans
- Database designs through to schema DDL (data definition language)
- Application system designs possibly through to computer program source code libraries
- Technology environment designs through to actual installation artifacts

But within this context, would not include:

- Actual databases with data records of employees, invoices, products, and customers
- Executing application systems
- Operating systems and other systems software such as DBMS and Web browsers
- Telecommunications Networks
- Computers

These are not metadata because they are "real," while the previous list represents artifacts about the reality. But once the information system is executing, metadata may be created that describes the characteristics of the operating environment. That class of metadata would include for example:

- Computer system execution schedules
- Computing resource consumption requirements
- Quantity of records in particular files
- Quantity of users by time of day for particular processes
- Job completion and/or error messages

Metadata is not just related to information systems. Rather it embraces all "data" at the meta level as is attested to by the following references:

| Title | URL |
|---|---|
| What every CIO Needs To Know About Metadata | http://www.cio.gov/Documents/metadata_cio_knowledge_Feb_1999.html |
| Metadata Primer | http://www.lic.wisc.edu/metadata/metaprim.htm |
| The Value of Metadata | http://www.fgdc.gov/publications/documents/metadata/metabroc.html |
| What Metadata Is and Why It is Important | http://tiwrm.hpcc.nectec.or.th/TECHNOLOGY/METADATA/WISCLINC/qa-meta.html |
| Frequently Asked Questions About Metadata | http://www.dublincore.org/resources/faq/ |
| Managing and Searching Data with Metadata | http://www.infoloom.com/gcaconfs/WEB/chicago98/hhrath.HTM |
| UK Metadata Framework | http://www.ogc.gov.uk/km/reports/metadata_v1.0.pdf |
| Practical Guide to Enterprise Architecture, Version 1.1. | http://www.cio.gov/documents/bpeaguide.pdf |
| Enterprise Architecture Framework, Version 1.1, September 1999 | http://www.cio.gov/documents/fedarch1.pdf |

## 2.0 Why Is Metadata Important

It may seem intuitively obvious why metadata is important. After all, it represents the specifications of the items that are being constructed, that is, the databases, application systems the networks, and the computer environments. But in a computing environment, metadata is far more critical than a traditional set of artifacts associated with the construction of a building. Computing environment metadata must be *live*.

If, for example, someone is building a house, then the requirements of the house might be a document. So too would be the blueprints and all the detailed drawings necessary to construct the house. In this paper's use of the term, these are metadata. Once a document is finished, it is reviewed, approved, and essentially remains as it last existed, forever. Once the owner approves a document, the effort continues to the next phase or perception. The approved documents are mainly consulted (and often only) for reference. If something is changed further down the building process the up-stream documents are seldom updated. They are not updated because it isn't critical to do so. Once the house is built, it's metadata either never becomes the property of the homeowner, or is put away in some drawer to gather dust. Houses seldom change. But when they do change, the process of investigating its existing structure is commonly simple and straightforward. In short, there is almost never a need to consult the house's blueprints. This is not the case with a very complex building, or with a suite of buildings. Their blueprints are consulted, but rarely updated. Computing environments, however are different for the following reasons:

- Computing environments constantly change during all stages of specification through development.
- Computing environments constantly change during the entire production life
- Computing environments changes are commonly pervasive and require understanding of and changes to components within the entire specification set.
- Computing environment metadata can be used as access mechanisms to the underlying components.

In considering the first point, changes during all stages of specification through development, a housing development's plans and documents, from the earliest land use sketches through to selling completed houses are supposed to represent the housing development accurately. For the initial land-use map, prior to the granting of zoning, the land area may be represented in broad generalizations such as residential areas, commercial shopping areas, commercial office areas, and part/recreation areas. Once the entire development is complete, it should be definitely possible to recognize the correspondence between this initial land-use map and the finished product. If, when reviewing the final development, the commercial office area that was in the land use proposal is missing, then either the initial plan is “broken,” or the final community is broken. If during the land use zoning process the commercial office are was deleted by the developer or by the zoning authority, then the plans would be approved subject to their correction and final submission. So what represents “broken-ness” here is that something that was explicitly listed as present is now missing, or that something that should have been listed as present is now there. Detailing is not necessary in all levels of specifications, either in a housing development nor in a computing environment. In the land use plan, it would therefore be inappropriate to either ask the question or be able to provide an answer to the following: “How many 120 volt receptacles will there be?” Similarly, in a computing environment, the names of the computer languages, the quantity of modules, or even the choice of a DBMS brand would be irrelevant questions with respect to an overall systems architecture diagram. But, if the on-line access component were deleted or added, then a case could me made that the systems architecture was indeed broken. And even if it were added at a later date, then the architecture would be broken.

At each subsequent stage of either the housing development or the computing environment, the perception of what was actually going to be built would be refined and more detailed. Each such refinement must be faithful to is antecedent layer. If the refinement cannot be traced back, then clearly one or the other is broken and must be fixed.

In the end then, there should be one completely integrated set of specifications, from initial requirements through to completed effort, each unfolding or being more refined than its predecessor with no surprises. While this is “nice” in a housing development environment it might not be absolutely necessary. But in a computing environment, it is essential.

For example, suppose there is a mapping between a set of cells on forms and view-columns (and thus, columns of tables in a database) in advance of building an information system. Suppose further the inventory of forms and form cells is cross-referenced and recorded onto its own miniature MS/Access database that was then retired into the grave-yard of completed deliverables. Then, suppose the changes occurred to both the target data model‘s table structures and column names. That of course would only be likely about 100% of the time. If the cross-reference MS/Access database is then retrieved, it will likely appear to be incorrect. Whether or not it was incorrectly done, it is now useless for any forms-based data entry screen building. The entire cross reference process has to be re-accomplished.

In this particular example, which was not just theoretic, there were 1500 columns across about 100 tables and about 1400 form cells. The re-analysis and re-mapping took about one staff month. The persons who had previously done the work were nowhere to be found. That complete re-analysis effort could have been prevented if there had been a complete integration among all the deliverables within the computer system’s metadata specification environment.

Examples like this are not the exception. Rather, they seem to be the rule. Deliverables such as tables and columns by business function, tables and columns based on specific security concerns and the like are all created through processes that involve multiple tools. For example, a data model may be created in a CASE tool, its tables and columns extracted into spreadsheets, which are then combined with explanatory text and diagrams, published and finally stored away as if there will never be another change. There always are changes. And half are remembered to be changed and of the half changed, they are changed incompletely. In short, it's just a big mess. To avoid these classes of problems, all the metadata from all the components of the specification must be integrated through all the metadata levels to exhibit consistency and integrity.

In considering the second point, changes during all stages of production, these can be from simple to radical. Are all these changes documented from their requirements through all stages of specification and then through development and into production? Few people would take even bets on that proposition. As a consequence, as a system moves through time in its production and evolution cycles, its specification metadata becomes not only more out of date, but also more in conflict and more irrelevant when compared to the actual executing application system. Exacerbating that even further is the almost certain likelihood that different groups of staff will have accomplished the modifications employing different techniques and styles for all their work. Change cycles become longer and are more risk-prone because the likely true effect of any change becomes less and less known. Sometimes the risk is so great that a decision is often made to completely re-systemized an entire area. Pity the poor user's request for a cost estimate for just a small change when that decision is made. To avoid these classes of problems, the completely integrated specification metadata must be available to production system change teams so they can make accurate assessments of any required changes.

In considering the third point, that is, that computing environments changes require understanding of and changes to components within the entire specification set, as the system becomes older and because the specification set of metadata is more out of synch with the production system, the effort required to make changes grows over time. In the first point, the specification metadata for form cell mappings was out of synch with the database. Then, over time the database, in possibly different ways. moves even further out of synch. Thus, when there is a need to understand the relationship between a form and the database design, both sets of metadata (specification and production-change) must be fixed before any real analysis can occur in support of configuring a change.

As to the last point, that within a computing environment, the metadata can act as access mechanisms may be unique within Information Technology. If a set of house blue prints exists, it is virtually impossible to employ sections of the blue prints as access mechanisms to the underlying components such that you finally find yourself in the actual room of the house. In computing environment metadata, access might first be to a list of systems then once one is selected access proceeds to a diagram of subsystems and interrelationships, then to a particular subsystem then to a list of programs, then to the structure of the program, and finally to the program's source code, which analogously is the actual room. In a more practical example, if there was an end-to-end connection among the metadata from the form cells through to the view-elements to the columns of tables, then if a column of a table was eliminated then some form cell would immediately be "uncovered," or even better, the metadata software system might not allow the removal of the column because it is connected to a form cell.

Further, if the specification metadata was completely integrated with the production-change metadata, then an answer would be readily available as to how much it would cost and how long it would take to make a change in the system's specification. To achieve all these metadata interactions requires both a sophisticated metadata environment and the foresight to see that such an environment is a worthwhile objective to achieve.

## 3.0 Enterprise Architecture Framework

The United States Federal Government has created an enterprise architecture framework, that is commonly known as the Federal Enterprise Architecture Framework, or FEAF. Web references to the FEAF are included in the table at the end of Section 1. In addition to describing an overall layered approach to the accomplishment of Information Technology projects, the architecture inferentially identifies a set of infrastructure work products that must be built to have IT success. The infrastructure product set is spread across four levels, which are:

- Architectural Models
- Business and Design Models
- Design Architectures for data, applications and technology, and
- Detailed exposition and actual implementations of the data, applications, and technology models

Figures 1 through 4 depict each level of the framework. Each level has a very high-level set of infrastructure products. And, each of these infrastructure products has both an internal design of varying complexity and relationships within and between other infrastructure products. These four levels were first specified and are eloquently detailed in Steven Spewak's book, *Enterprise Architecture Planning: Developing a Blueprint for Data, Applications and Technology*. (John Wiley & Sons, 1992).

Figure 4 is the Zachman Framework (originally called the Framework for Information Systems). The Zachman framework has always been viewed as being unbound and generalized. The U.S. Federal Government's CIO council took the Zachman framework and reworked it into a Level IV for its Federal Enterprise Architecture. The row titles remained but the column titles and cell contents were changed. The CIO Council also dropped the sixth row, operating enterprise from the Zachman Framework. In short, the Federal Enterprise Architecture is "Zachman" in name only. Essentially, the resulting Level IV consists of three columns and five rows. This is depicted in Figure 5. Each column infers a major category of work products, and the set of all products is related to:

- Data Architectures,
- Application Architectures, and
- Technology Architectures

The enterprise architecture's five rows represent an unfolding set of detail through which the column's product set is both viewed and transformed in order to be realized as an operating enterprise. To be complete, there are three more columns to the Zachman Framework: People, Time, and Motivation. These columns were summarily described in the enterprise architecture. Table 2 that follows Figure 5 enumerates the six columns across its five rows.

The four level, three column and five row strategy for applying the framework is illustrated in Figure 6. The top three levels (Planning Initiation, Business Modeling & Current Systems and Technology, and Data Architecture & Application Architecture & Technology) accomplished. Then a Zachman-like, detailing of Level IV through it's three columns (Data Architecture, Application Architecture, and Technology Architecture) and five perspective rows

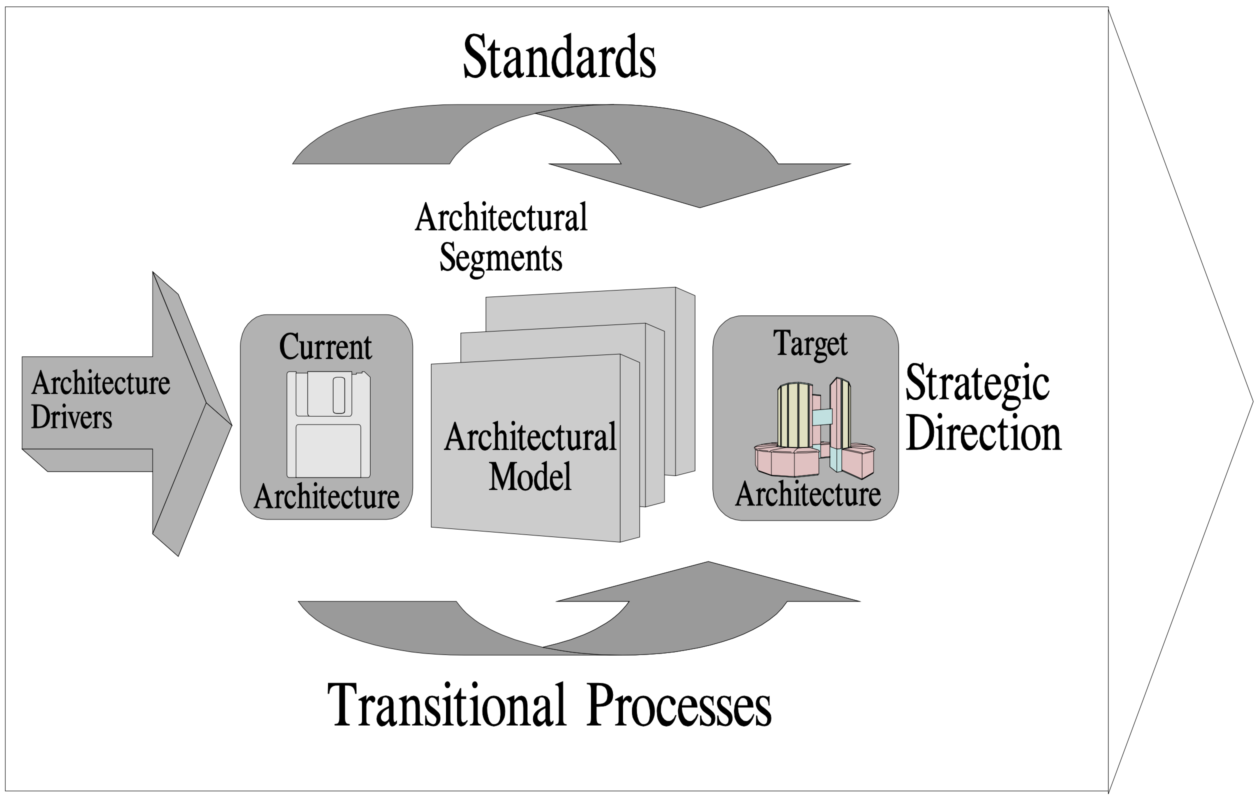



**Figure 1**. Enterprise Architecture Framework Level I

(Planners view, …, Subcontractors View) are accomplished.

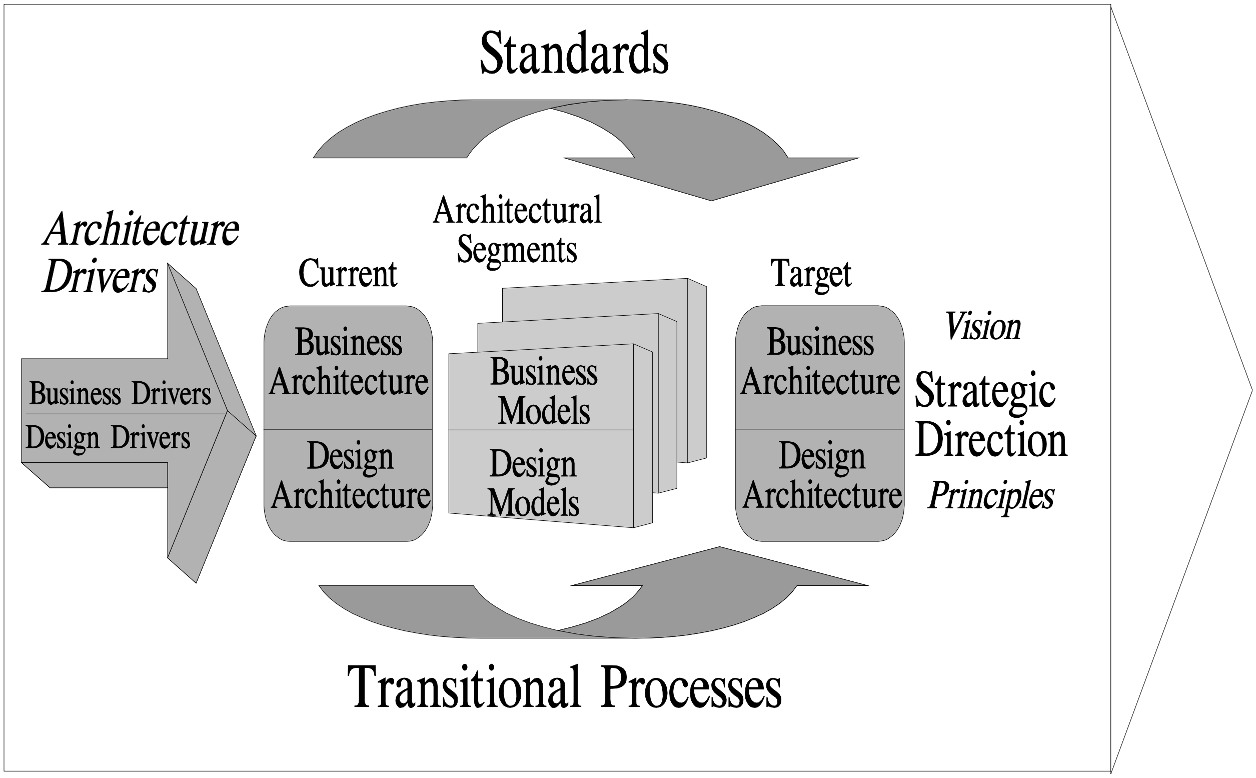



**Figure 2**. Enterprise Architecture Framework Level II.

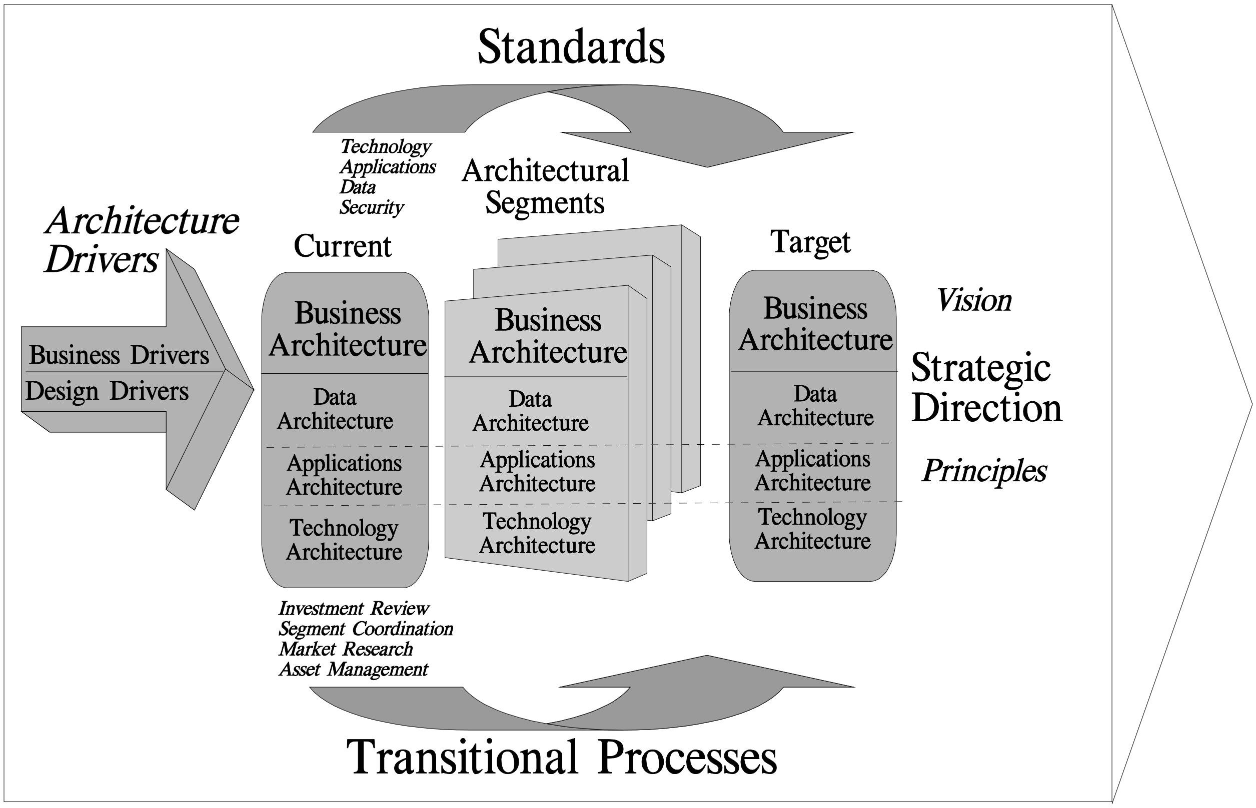



**Figure 3**. Enterprise Architecture Framework Level III

| The Zachman Framework | | | | | | |
|---|---|---|---|---|---|---|
| | **Entities (what)** | **Activities (how)** | **Locations (where)** | **People (who)** | **Time (When)** | **Motivation (why)** |
| Planner | | | | | | |
| Owner | | | | | | |
| Designer | | | | | | |
| Builder | | | | | | |
| Sub-contractor | | | | | | |

**Figure 4**. Enterprise Architecture Framework Level IV

| Perspectives | Data Architecture | Applications Architecture | Technology Architecture |
|---|---|---|---|
| **Planner's View Objectives/Scope** | List of Business Objects | List of Business Processes | List of Business Locations |
| **Owner's View Enterprise Model** | Semantic Model | Business Process Model | Business Logistics System |
| **Designer's View Information Systems Model** | Logical Data Model | Application Architecture | System Geographic Deployment |
| **Builder's View Technology Model** | Physical Data Model | System Design | Technology Architecture |
| **Subcontractor's View Detailed Specifications** | Data Definition | Programs | Network Architecture |

**Figure 5**. Enterprise Architecture Framework, Level IV.

| Federal Enterprise Framework Cell Contents (Table 2) | | | |
|---|---|---|---|
| **Perspectives** | **Data Architecture (what)** | **Application Architecture (how)** | **Technology Architecture (where)** |
| **Planner** | List of Business Objects. A list of objects (or things, or assets) in which the enterprise is interested. The list is a fairly high level of aggregation. The model defines the scope, or boundaries, of the models of objects significant to the enterprise (i.e., the rows beneath it). | List of Business Processes. A list of processes or functions that the enterprise performs, or the transformation of enterprise inputs into outputs. The list is a fairly high level of aggregation. The model defines the scope, or boundaries, of the models of processes the enterprise performs (i.e., the rows beneath it). | List of Business Locations. A list of locations in which the enterprise operates. The list is a fairly high level of aggregation. The model defines the scope, or boundaries, of the models of locations that are connected by the enterprise (i.e., the rows beneath it). |
| **Owner** | Semantic Model. A model of the actual enterprise objects (i.e., things, assets) that are significant to the enterprise. Typically, the semantic model would be represented as an entity/relationship model and would be at a level of definition expressing concepts (i.e., terms and facts) used in the significant business objectives/strategies that would later be implemented as business rules. | Business Process Model. A model of the actual business processes that the enterprise performs, independent of any system or implementation considerations and organizational constraints. It can be represented as a structured methods-style model expressing the business transformations (processes) and their inputs and outputs. | Business Logistics System. A model of the locations of the enterprise and their connections (i.e., voice, data, post or truck, rail, ship, etc.). It would include identification of the types of facilities at the nodes like branches, headquarters, warehouses, etc. |
| **Designer** | Logical Data Model. A model of the logical representation of the objects of the enterprise about which it records information, in either automated or nonautomated form. It would be represented as a fully attributed, keyed, normalized entity relationship model reflecting the intent of the semantic model. | Application Architecture. A model of the logical systems implementation (manual and/or automated) supporting the business processes. It expresses the human and machine boundaries. The model could include the controls and mechanisms, as well as the inputs and outputs to the logical systems representations of the system functions/processes. | System Geographic Deployment Architecture. A logical model of the system implementation of the business logistics system depicting the types of systems facilities and controlling software at the nodes and lines (e.g., processors/operating systems, storage devices/DBMS, peripherals/drivers, lines/line operation systems, etc.). |

| Federal Enterprise Framework Cell Contents (Table 2) | | | |
|---|---|---|---|
| **Perspectives** | **Data Architecture (what)** | **Application Architecture (how)** | **Technology Architecture (where)** |
| **Builder** | Physical Data Model. A technology constrained, or physical representation of the objects of the enterprise. The representation style of this model would depend on the technology chosen for implementation. If relational technology is chosen, this would be a model of the table structure required to support the logical data model in a relational style model. In an object-oriented notation, this would be a class hierarchy/association style model. | Systems Design. Technically, this would not be considered a model but a design. At a high level of abstraction, it would be a structure chart and in its detail, action diagram-style expressions that would constitute the implementation of the logical systems, or application architecture. In object-oriented notation, this would be the methods and their realization. | Technology Architecture. The physical depiction of the technology environment for the enterprise showing the actual hardware and systems software at the nodes and the lines and their systems software, including operation systems and middleware. |
| **Subcontractor** | Data Definition “Library or Encyclopedia”. The definition of all the data objects specified by the physical data model and would include all the data definition language required for implementation. | Programs “Supporting Software Components (i.e., Operating Systems)”. The programs derived from the action diagram style or object-style specifications for the implementation. Given the appropriate engineering design, these could become the prefabricated components that could be assembled into more than one implementation. | Network Architecture. The specific definition of the node addresses and the line identification. |

| Federal Enterprise Framework Cell Contents (Table 2) | | | |
|---|---|---|---|
| **Perspectives** | **People (who)** | **Time (when)** | **Motivation (why)** |
| **Planner** | List of Organizations. Important to the Business A list of organizations to which the enterprise assigns responsibility for work. The list is at a fairly high level of aggregation. It defines the scope or boundaries of the models (i.e., the rows beneath it). | List of Events. Significant to the Business A list of events to which the enterprise responds relative to time. The list is at a fairly high level of aggregation. It defines the scope or boundaries of the models (i.e., the rows beneath it) of time significant to the enterprise. | List of Business Goals/Strategies. A list of major business goals, objectives, strategies, or critical success factors significant to the enterprise relative to motivation. The list is at a fairly high level of aggregation. It defines the scope or boundaries of the models (i.e., the rows beneath it). |
| **Owner** | Work Flow Model. The model of the actual enterprise allocation of responsibilities and specification of work products. Typically, an organization chart expresses the allocation of responsibilities, but other supporting documents describe the work products. To be complete, the organization chart would have to be supplemented with work products (e.g., control work, coordination work, and operational work) and the originating and receiving organization units identified. | Master Schedule. A model of the business cycles comprised of an initiating event and an elapsed time. There are two typical notations for expressing points in time and lengths of time, P.E.R.T. charts and the Senge or systems thinking models. Senge models are not definitive in relation to the length of the time cycle or sequence. | Business Plan. A model of the enterprise business objectives and strategies that constitute the motivation behind enterprise operations and decisions. Although there has been considerable focus on management theory in academia, no commonly accepted notation for the motivation concepts exists. |
| **Designer** | Human Interface Architecture. The logical systems expression of work flow, which includes specification of the roles of the responsible parties: management, administration, knowledge-worker, engineering, marketing, etc., and the logical specification of work products (e.g., voice, text, graphics, video, etc.). | Processing Structure The logical systems specification of points in time (i.e., systems events) and lengths of time (i.e., processing cycles). This model describes the system events that trigger transition from one valid state (i.e., point-in-time) to another and the dynamics of that transition cycle. This model is represented in the notation of an entity life history diagram (from the SSADM methodology that originated in the U.K.) or in the notation of a an object-oriented Harel state chart. Petri Nets are also used to express time sequence aspects. | Business Rules. A logical model of the business rules of the enterprise in terms of intent and constraints. No commonly accepted notation currently exists for business rules. |

| Federal Enterprise Framework Cell Contents (Table 2) | | | |
|---|---|---|---|
| **Perspectives** | **People (who)** | **Time (when)** | **Motivation (why)** |
| **Builder** | Presentation Architecture. This is the physical expression of enterprise work flow including specific individuals and their ergonomic requirements and work product presentation format. | Control Structure. The physical expression of system events and physical processing cycles, expressed as control structure, passing "control" from one processing module to another. | Rule Design. This is a physical specification of the business rules. The rules are not presently factored out from their implementations and, therefore, are shown as cardinally and optionally in the data models, as procedural code, or as policy specification. However, historically, there have been "inference engine"-style technologies that allow expression of rules quite independent from data and logic, and the tools in which these ideas persist may influence the general marketplace with their formalisms. |
| **Subcontractor** | Security Architecture. The out-of-context specification of work flow would be identification of the individuals accessing the system and the work or job they were authorized to initiate. | Timing Definition. The definition of interrupts and machine cycles. | Rule Specification. This will be the out-of-context specification of the business rules. |

| **Level I** | **Planning and Initiation** | | |
|---|---|---|---|
| **Level II** | **Business Modeling** | **Current Systems and Technology** | |
| **Level III** | **Data Architecture** | **Applications Architecture** | **Technology Architecture** |
| **Implementation/Migration Plan for Achieving Level III, that is, Level IV** | | | |
| **Perspectives** | **Data Architecture** | **Applications Architecture** | **Technology Architecture** |
| **Planner's View Objectives/Scope** | List of Business Objects | List of Business Processes | List of Business Locations |
| **Owner's View Enterprise Model** | Semantic Model | Business Process Model | Business Logistics System |
| **Designer's View Information Systems Model** | Logical Data Model | Application Architecture | System Geographic Deployment |
| **Builder's View Technology Model** | Physical Data Model | System Design | Technology Architecture |
| **Subcontractor's View Detailed Specifications** | Data Definition | Programs | Network Architecture |

**Figure 6**. Enterprise Architecture Framework, Levels I, II, III, IV

The Zachman Framework, published in 1987, has remained fundamentally unchanged. Zachman never detailed the work products into a comprehensive set, nor created a detailed methodology through which the Zachman Framework for Information Systems could be applied rigorously and unambiguously by multiple teams across the enterprise. Finally, Zachman never created a meta entity schema of the work products that showed all their necessary interrelationships. Thus, the Zachman Framework remains very generalized and unbound.

If the Zachman Framework is strictly followed, its inferred infrastructure product set represents neither what is needed nor what is possible. Figure 7 shows the allocation of a distilled set of about 120 "failure factors" (transformed to percent) that were mined from a collection of 13 multi-hundred million dollar U.S. Government's IT system failure. The table shows that only 10% of the failure factors are addressed by the Zachman Framework.

| **Zachman Information Systems Architecture Framewok** | | | | | | |
|---|---|---|---|---|---|---|
| | **Data** | **Function** | **Network** | **People** | **Time** | **Motivation** |
| **Scope** | 2 | 1 | None | None | None | 5 |
| **Enterprise Model** | 1 | 1 | None | None | None | None |
| **System Model** | None | None | None | None | None | None |
| **Technology Model** | None | None | None | None | None | None |
| **Detailed Representa-tions** | None | None | None | None | None | None |
| **Functioning Enterprise** | None | None | None | None | None | None |

**Figure 7**. Allocation of the GAO Information Technology Critical Success Factor Issues.

To resolve this inadequacy or insufficiency problem, the U.S. Federal government's guidance explicitly states that alternate frameworks may be employed. Figure 8 portrays an alternative framework, called the Knowledge Worker Framework. It more fully addresses the needs of enterprise architecture projects. This framework differs from the Level IV of the enterprise architecture in the following ways:

- The Knowledge Worker Framework has six full columns. It added Mission (motivation), Organization (people), and Business Event (time). The Mission column is clearly a richer than just business rules. The Business Event column represents the time and business cycle intersection between human activity (Function), and automation activities (Business Information Systems). These columns are strictly sequenced, left-to-right, as follows: Mission, Database Objects, Business Information Systems, Business Events, Business Functions, and Business Organization.

- The enterprise architecture column, Application Architecture, has been divided into two columns, Business Information Systems, and Business Event.

- The Business Function column exclusively represents the human functions that are essential to model and set into motion.

- The enterprise architecture column, Data Architecture, has been changed as follows: First, it explicitly meets the needs of a wide variety of data architecture classes that must be present in modern system environments. Second, it models the essential resources of the enterprise and their life cycles around which databases and information systems are identified, designed, implemented and managed. Third, it restores the persistent data object orientation that was common to databases and DBMSs of the Sixties and Seventies that was removed by the drive to relational DBMS, but has now been restored by modern ANSI SQL standard compliant DBMSs. Fourth, it includes well defined integrity constraints and state-based object behavior is essential to a well-ordered database environments. These facilities are all properly the domain of the persistent data layer not the end-user language dependent application layer. These Sixties and Seventies integrity and behavior facilities are now being implemented by modern, ANSI standards conforming, object oriented SQL DBMSs.

- The Knowledge Worker Framework has a full sixth row, operations that represents the instantiated metadata artifacts from an operating environment and also the feedback from these operating systems and databases to support "live" metadata environments..

The Knowledge Worker Framework is comprehensively supported by a complete set of work products, a detailed methodology, a meta entity schema, and a metadata repository/CASE system.

| Knowledge Worker Framework | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | Man-Machine Interface | | | | | |
| | | Machine | | Interface | Man | | |
| **Deliver-ables** | **Mission** | **Database Object** | **Business Information System** | **Business Event** | **Business Function** | **Organization** | |
| **Scope** | List of business missions | List of major business resources | List of business information Systems | List of interface events | List of major business scenarios | List of organizations | |
| **Business** | Mission hierarchies | Resource Life Cycles | Information sequencing and hierarchies | Event sequencing and hierarchies | Business scenario sequencing and hierarchies | Organization charts, jobs and descriptions | |
| **System** | Policy hierarchies | Specified data models and Identified Database objects | Information system designs | Invocation protocols, input and output data, and messages | Best practices, quality measures and accomplishm ent assessments | Job roles, responsibilities, and activity schedules | |
| **Technology** | Policy execution enforcement | Implemented data models and Detailed Database Objects | Information systems application designs | Presentation layer information system instigators | Activity sequences to accomplish business scenarios | Procedure manuals, task lists, quality measures and assessments | |
| **Deploy-ment** | Installed business policy and procedures | Operational data models | Implemented information systems | Client & server windows and/or batch execution mechanisms | Office policies and procedures to accomplish activities | Daily schedules, shift and personnel assignments | |
| **Operations** | Operating business | Application view data model | Operating information systems | Start, stop, and messages | Detailed procedure based instructions | Daily activity executions, and assessments | |

**Figure 8**. The Knowledge Worker Framework.

The resultant Level IV knowledge worker framework is well suited to provide the following IT assets:

- The essential missions that define the very existence of the enterprise, and that are the ultimate goals and objectives that measure enterprise accomplishment from within different business functions and organizations.

- The procedures performed by groups in their achievement the various missions of the enterprise from within different enterprise organizations.

- The organizational mission accomplishments with databases, information systems and functions

- The key Resources (facilities, materiel, staff, etc.) and how they are sequenced, interrelated, and how are they supported through databases and information systems.

- The information (a.k.a. query results or reports) needed by various organizations in their functional accomplishment of missions and what databases and information systems provide this information.

- The data needed by functional proponents, how it is defined within data architectures and databases and how and where are those databases are deployed and then used by business information systems in support of mission accomplishment.

- The context independent semantic templates of data elements and how these are configured into models of data (the consequence of policy execution) determined as needed by functional experts in support of enterprise missions, and how are these specified data model requirements configured into implemented databases that ultimately operate within various organizations as they perform the functions needed by enterprise missions.

- The business information systems, where they are, how they are related to mission, organization, function, databases, and the impact on these business information systems when policy (a.k.a., data) is required or changed.

- The identification, allocation, and scheduling of database and information system projects within the context of Resource Life Cycles.

- The identification, estimation, and then monitoring of database and business information system projects from within the context of well established metrics, and templates work breakdown structures, and deliverables. The resulting project schedules are then able to be accomplished in a business resource defined sequence to best achieve enterprise missions through business functions and organizations.

The Knowledge Worker Framework cannot however just replace Level IV. That is because the Knowledge Worker Framework does not contain Technology Architecture. It is too simple, however, to just add another column. Technology today contains a complex network of components that includes at the very minimum:

- Computers for personal, server, mid-range, and mainframe
- Operating systems for all classes of computers, and all other kinds of systems software such as telecommunications monitors, job schedulers, backup software, and the like.
- Networks for LAN, WAN, etc.
- Intranets
- Internets
- Database management systems
- Office and application software suites
- Data transport layers such as XML and associated software

Technology architecture deserves a framework of its own. The rows (scope, business, etc) would likely be the same as the Knowledge Worker Framework, and some of the columns might be the same as well (Mission and Organization) but the remaining columns need to be created and populated appropriately. The two frameworks, Knowledge Worker and Technology would be intersected through their commonly employed metadata repository. This entire process is described at in the paper, *Frameworks, Metadata Repository, and the Knowledge Worker* (http://www.wiscorp.com/tdan.html.)

The net effect of these changes is that the goal and intent of the enterprise architecture is more completely accomplished. Further, carrying out recommendations of this type is encouraged in the document, *A Practical Guide to the CIO Council's Enterprise Architecture* (February 2001).

The reasons for substituting the Knowledge Worker Framework for the Level IV Zachman Framework and for installing a modified product set into all six columns and rows of the framework are telling. An analysis of 13 $100+ million GAO Federal IT project failures pointed to over a 120+ reasons for IT system failure. When those reasons were allocated to the Zachman Framework and to the Knowledge Worker Framework, the Zachman Framework, if followed as specified, addresses less than 10% of the reasons while the Knowledge Worker Framework addresses all 100%. Figure 9 presents the percentage distribution of those IT failure reasons addressed by the Knowledge Worker Framework. Two observations are very noteworthy. First, almost none of the reasons reside within the shaded cells that are the IT cells. That means that 95% of the reasons why IT systems fail occur outside IT. Second is that 50% of the reasons why IT systems fail occur actually after the IT system has been delivered.

When the Knowledge Worker Framework is substituted for the Zachman Framework not only is the resulting set of metadata products more complete with respect to the requirements of database application environments, they also match more precisely the work products required by the text by Steven Spewak, Enterprise Architecture Planning: Developing a Blueprint for Data, Applications, and Technology. This book is a key foundation for enterprise architectures.

Finally, The Knowledge Worker Framework is supported by a complete metadata repository and CASE tool, courses, workshops, seminars, and a complete database project methodology. Figure 10 illustrates the Federal Enterprise Framework that is described throughout the rest of this paper.

| Knowledge Worker Framework | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | Man-Machine Interface | | | | | |
| | | Machine | | Interface | Man | | |
| **Deliver-ables** | **Mission** | **Database Object** | **Business Infor-mation System** | **Business Event** | **Business Function** | **Organ-ization** |
| **Scope** | 5 | 2 | 3 | 1 | 3 | 4 |
| **Business** | 5 | 3 | 2 | 1 | 6 | 6 |
| **System** | 3 | **2** | **2** | **1** | 12 | 8 |
| **Tech-nology** | 1 | **0** | **0** | **0** | 8 | 6 |
| **Deploy-ment** | 0 | **0** | **0** | **0** | 5 | 5 |
| **Oper-ations** | 0 | **0** | **0** | **0** | 3 | 3 |
| **Col. Totals** | **14** | **7** | **7** | **3** | **37** | **32** |

***Note:*** All numbers expressed as Percent allocations of errors to cells ...12
Gray cells are IT Cells

**Figure 9**. Allocation of the GAO Information Technology Critical Success Factor Issues

<table>
<tr><td>Level I</td><td colspan="6">Planning and Initiation</td></tr>
<tr><td>Level II</td><td colspan="3">Business Modeling</td><td colspan="3">Current Systems and Technology</td></tr>
<tr><td>Level III</td><td colspan="2">Data Architecture</td><td colspan="2">Applications Architecture</td><td colspan="2">Technology Architecture</td></tr>
<tr><td colspan="7">Implementation/Migration Plan for Achieving Level III, that is, Level IV</td></tr>
<tr><td>Perspectives</td><td>Mission</td><td>Database Object</td><td>Business Info Systems</td><td>Business Event</td><td>Business Function</td><td>Business Organiza-tion</td></tr>
<tr><td>Scope</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>Business</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>System</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>Technology</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>Deployment</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>Operations</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>

**Notes**: The next level of detailing to the 36 cells above is in Figure 8. The detailing of Technology Architecture is accomplished in a Technology Architecture framework that is outside the scope of this document.

**Figure 10**. Enterprise Architecture Framework, Levels I, II, III, IV

## 4.0 Metadata Work Products

An enterprise architecture project, as a project, requires two very different product sets. The first product set, the *infrastructure product* set, is required for the work products that result from analysis, design, implementation, and maintenance efforts. Collectively, these work products essentially represent a project's infrastructure through which the project is planned, accomplished, managed, evolved, and reported. More simply, these work products *are* the project.

The second product set, *external products*, are the project's actual products, that is, the database applications, the actual databases, policies, and procedures. These, in turn, manage the "real" data represented by the enterprise's mission, that is, real production data.

Figure 11 illustrates the critical components of a project environment and their interrelationships. There are three major process activities: Analysis and Design, Build Computing Environment, and Operate Computing Environment. In Figure 11, Analysis and Design activities are performed according to tasks from a project work plan via one or more infrastructure templates. The work that results should be stored in the metadata repository and is made visible as a project external product. As an example, an infrastructure product template might be an Erwin data model. The analysis and design task would be the creation of the logical data model. And the work result would be the set of ER diagrams, and if externalized, SQL schema DDL. What would be missing from the metadata repository would be the suite of data models as their metadata information would be locked within Erwin.

Then, during the effort to actually build the project's computing environment, the external work product is combined with a computing environment product with the result of creating a computing environment product. As an example, the SQL DDL would be "read" by the DBMS (e.g., Oracle) and the result is an Oracle database schema.

In the final major process activity, the computing environment is used by "end-users" to import/export, capture, update, and report the actual data.

Why the infrastructure product set is important is dramatically illustrated through the Standish Corporation's survey that showed that in the late 1990s, 85% or more of IT systems were significantly late, over budget, or outright failed, and the three biggest reasons were: 1) Incomplete Requirements, 2) Lack of User Involvement, 3) Lack of Resources. In contrast, when IT systems succeeded, the three reasons for success were: 1) User Involvement, 2) Executive Management Support, and 3) Clear Statement of Requirements. It is critical to note that none of these reasons for IT success and none of the reasons for IT failure are within the proper domain of IT. IT success is not, however, assessed against the initial goals of the effort, but almost always on the relevance of completed product even though IT almost never has any control over effort goals and goal evolutions. To be successful, IT must track, manage, evolve, and react to all the factors that comprise the IT environment.

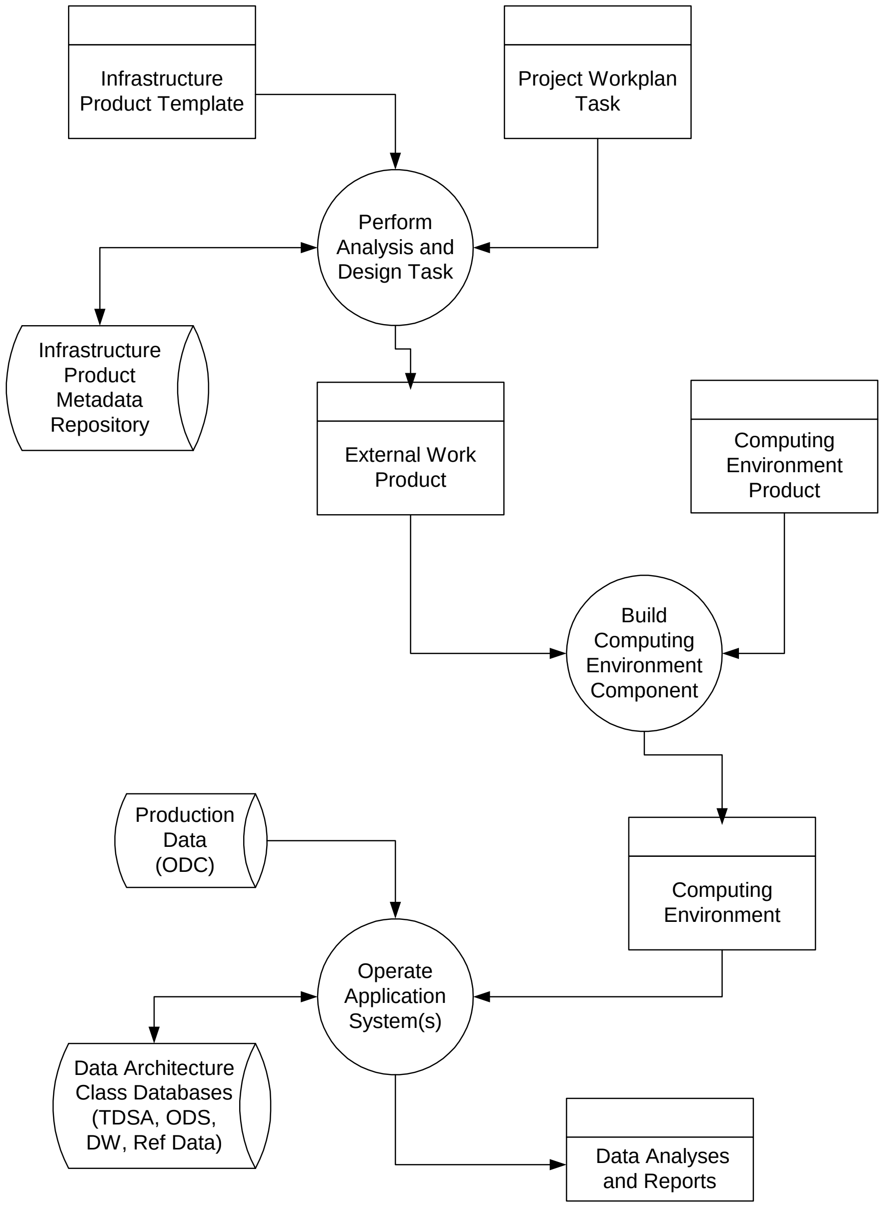



**Figure 11**. Database project work product flow.

Traditionally, infrastructure products are represented through documents. A small subset of these are listed in the table below, are good examples. Other products within the infrastructure product set include project plans and reports to oversight organizations.

Some, or aspects of the Infrastructure Product sets are somewhat automated, e.g., Microsoft Word, Excel, Microsoft project, Power Point presentations, and Erwin data models. In the main, however, almost all the significant content of all the existing set of Infrastructure Products are hard-locked into their contained format, that is, words in sentences in paragraphs within a Word document, or project tasks and resources within MS/Project, or finally, the detailed data model information within Erwin. Two problems immediately result. First, the significant content of each infrastructure product cannot be queried, updated easily and interrelated with other content in the same document. Secondly, and even more importantly, the product content of one infrastructure product cannot be interrelated with the product content of any other infrastructure product. The reason for these problems is simple. The infrastructure product content is locked into a proprietary format. Resolving this problem is equally simple. All infrastructure product content must be stored in a ANSI SQL Standard metadata repository database such that the content data is available for use for processing and reporting by products like Word, MS/Project, Erwin, and the like, and also that the content can be queried, updated easily, interrelated and reported through comprehensive database report writers.

Having the infrastructure product set within a database environment is no different in value or purpose than having production data in a database except for the fact that it is just one level removed. Without production data an enterprise cannot hope to effectively and efficiently know about or manage the mission of the enterprise. Similarly, without infrastructure work product metadata an enterprise cannot hope to effectively and efficiently know about or manage its IT projects.

As John Zachman so often says, "*Someday you are going to wish you had all those models, Enterprise-wide, horizontally and vertically integrated at an excruciating level of detail*." That "someday" probably occurred about 25 years ago when it first became possible to have a unified database of infrastructure work products. And that "someday you are going to wish" also probably occurred about 25 years ago when it became possible to have enterprise-wide IT systems and databases. Enterprises that acted on Zachman's admonition often succeeded at their IT efforts, and those that do not almost always failed. The GAO library is a graveyard of failure tombstones (a.k.a., reports).

| Doc # | Document Title |
|---|---|
| 1.1.3 | Reengineering and Planning Guide |
| 1.2.1 | Catalog of Current Data Requirements |
| 1.3.1 | Catalog of Required Data |
| 1.3.2 | Report on the Implications of Statutes and Regulations |
| 2.4.1 | Consolidated Report on Analysis of Implications and Recommendations for Agency Transition |
| 2.5.1 | Updated Data Model |
| 3.3.2 | Physical Model Structure and Distribution Alternatives Analysis |
| 3.6.3 | Supplemental Spreadsheet to Business Case: Revised Benefit-cost Analysis—Accelerated Schedule |
| 3.6.4 | Business Case: Benefit-Cost Analysis |
| 4.1.2 | Outline of Planned Changes to the Data Model |
| 4.1.3 | Logical Data Model Issues and Impacts |
| 4.2.2 | Logical Data Model Mapping |
| 4.5.1 | Privacy Impact Assessment |

A brief analysis of just one infrastructure product document, for example, the Privacy Impact Assessment (document 4.2.5 from the table above) is telling. An examination of that document shows that it contains a structure for all its contents, introductory materials that set the context, and then chapters of its significant content. Such a document should not be its definition and storage layer for its infrastructure content. Rather, it should be just the presentation layer, which, when traversed leads to its contents within the metadata repository. If the document is the definition and storage layer then its content can easily be lost and outdated. If, rather, the document, is merely the presentation layer then as the content is updated, its "appearance" within any document is always current and is just a presentation layer generation step away.

Across a large quantity of documents there may be conflicts among the different instances of "significant content." These conflicts are due to evolutions, changes in scope, and the like. Conflicts also occur because across time, the "what and where" of significant content actually is lost, or even worse, even when its location is known, its content is not updated to then make it consistent across all other documents.

A good example of the "what and where" in the privacy impact analysis document is its excellent exposition of critical business functions. Because that material appears to have been originally created for that particular document, it's natural to ask: Is it elsewhere? The same? Different? Evolved? Conflicting? The answer to all questions is yes.

Appendix D of the this very same document is a good example of critically important but outdated information. It contains a cross reference between entities, their attributes, and privacy requirements. Some of the entities have been dropped and others added. Attributes within some entities have been dropped, others added, and some modified. Because of this outdated material the entire document has suspect validity.

It is not uncommon for the significant content of one document to be related to the significant content of one or more other documents. The collection of all significant contents across all the documents then represents the essence of the infrastructure information about a project. If all this infrastructure knowledge is integrated and current and not conflicting, then too is the project. Conversely, if some or critical parts of the infrastructure is outdated and/or in-conflict, then so too is the project.

In contrast to infrastructure products, external products (see Figure 9) include actual application systems, databases, computer hardware, operational policies and procedures. Many of the external products are easier to be kept current and synchronized one with another because of their very nature and because technology requires them to be current or they will not "work." For example, if a database's schema is updated and that update is reflected in the DBMS, then the applications that use that database through the DBMS may automatically stop working because there may be a conflict. Required is that the application programs be recompiled.

Across the enterprise, however, if one database's schema and its application systems are migrated to a slightly different design, its peer level, but separate database schemas and related application systems are not automatically updated. So, while within a given bound set of database and application systems, the evolutions can generally be done in lock-step, across the enterprise, the evolution of the related sets of databases and application systems require that their overlapping design and implementation artifacts be known and managed. When this is not done, stove-pipe systems occur. To resolve these inter-system conflicts, an entire industry, Extract-

Transform-Load (ETL) was created. In a 1995 study, the United States Air Force estimated that it spends $175 million annually on ETL type efforts.

The immediate consequence of the a set of project infrastructure work products that possess internal and inter-product inconsistencies is that prior to any meaningful work, the infrastructure product set must be evaluated for adequacy, currency, and for complete integration with all other infrastructure work products. The reason is simple. If the “authors” of the materials have not kept these materials in lock-step with all other related infrastructure materials, then a project’s system cannot be designed and built because of the significant content conflicts. The first step any project implementor must then do is learn, discover, resolve, propose, and the fix all significant content errors across all the documents before *step-one* of any *real* work can begin. And, it is likely that during this effort that the implementor will discover conflicts that will invalidate schedule, work product set, and systems and databases that need to be built.

The product set of this modified framework have been allocated too all the six columns and six rows. This allocation is presented via a high level summary in the tables that follow.

| | |
|---|---|
| **Row (1,1)** | Scope |
| **Column** | Mission |
| **Metadata work product description** | Identified are the missions that represent the ultimate vision that a enterprise architecture project is to accomplish. If a mission is missing then the enterprise is deficient in some material way. There are infrastructure missions that relate to internal processes and activities and external missions that relate to the products produced and/or sold and to the clients or customers. |
| **Benefit to a Enterprise** | The list missions that represent the idealized accomplishments of the enterprise. |

| | |
|---|---|
| **Row (1,2)** | Scope |
| **Column** | Database Objects |
| **Metadata work product description** | List of the identified and defined enterprise resources that represent the assets of the enterprise which are created, manipulated, and dissolved in support of the mission of the enterprise.<br><br>These ultimately lead to the identification and development of the database objects that, in turn, provide proof that the enterprise's mission is being accomplished.<br><br>Identified and described data architecture classes that must be managed.<br><br>Identified and defined database domains (derived from the mission hierarchies) that then form the basis for the various database designs. |

| | |
|---|---|
| **Benefit to a Enterprise** | Identified are the essential resources that must be managed by a project. The set of all resources are the minimum and essential assets of the enterprise. The resources are cross referenced to mission to ensure both are complete.<br><br>Identified as well is the database topology that represents the infrastructure databases and those databases that must exist within agencies and/or providers.<br><br>Database domains, detailed from the mission hierarchy leafs represent the names of all the key data groupings that must be represented within the environment to then satisfy the missions. |

| | |
|---|---|
| **Row (1,3)** | Scope |
| **Column** | Business Information Systems |
| **Metadata work product description** | Each business information system serves to achieve a state change in one or more of the enterprise's resources. Each business resource may become one or more database objects. |
| **Benefit to a Enterprise** | Provides the information flow strategies across all agencies/providers. Provides the information flow within the environment, and finally the information flow that must be supported to the "outside world." |

| | |
|---|---|
| **Row (1,4)** | Scope |
| **Column** | Business Events |
| **Metadata work product description** | List of events significant to the business including related calendar and/or business cycles. The list is at a fairly high level of aggregation. It defines the scope or boundaries of the models (i.e., the rows beneath it) of time significant to the enterprise.<br><br>List of Interface events between business functions and business information systems. |
| **Benefit to a Enterprise** | The business events provide the business operations context within which the business information systems operation in support of the various missions.<br><br>The relationships between business events, cycles, and calendars and their "serviced" business functions provide a cross check that every business information system is in support of one or more business functions. |

| | |
|---|---|
| **Row (1,5)** | Scope |
| **Column** | Business Function |
| **Metadata work product description** | The list of human-based functions that are accomplished in support of accomplishing the mission of the enterprise. There may be multiple functions that are generally equivalent but are performed by different organizations. Includes relationships between business functions and organizations. |
| **Benefit to a Enterprise** | The business function inventory provides a clear picture of the human activities that are being performed within the enterprise. As the essentially same function is performed by differently organizations the details of each can be examined to determine the relative efficiency of each. |

| | |
|---|---|
| **Row (1,6)** | Scope |
| **Column** | Organizations |
| **Metadata work product description** | List of organizations and organization hierarchies important to the business to which the enterprise assigns responsibility for work. The list is at a fairly high level of aggregation. It defines the scope or boundaries of the models (i.e., the rows beneath it). Includes relationships between organizations and business functions. |
| **Benefit to a Enterprise** | The identification of the agencies, organizations are necessary so that he appropriate allocations can be made of information needs, business functions, and business information systems and resources to assure that the coverage is both the minimum necessary and appropriate. |

| | |
|---|---|
| **Row (2,1)** | Business |
| **Column** | Mission |
| **Metadata work product description** | Mission hierarchies and all business terms and definitions to be supported by the environment. These sets of descriptions may comprise 10 to 15 pages of hierarchically organized text. |
| **Benefit to a Enterprise** | The detailed set of missions and their descriptions represent the hierarchy of the idealized accomplishments of the enterprise are the bedrock against which the entire project environment is determined, designed and built.<br><br>From the missions, the data requirements (via database domains and then database objects) are determined. Against the missions, the organizations and the various functions within the organizations are identified interrelated. |

| | |
|---|---|
| **Row (2,2)** | Business |
| **Column** | Database Objects |
| **Metadata work product description** | The resource life cycles are detailed. Each resource life cycle node represents a major value state for that resource. The data across the entire resource generally conforms to a subject area database.<br><br>The enumeration of the resources are the enterprise objects. The various resource life cycles are interconnected as may be appropriate. Additionally, the major databases and business information systems that are known are interconnected to the resource life cycle nodes. Collectively this produces a complete inventory of all IT database and system assets including their use within the enterprise. |
| **Benefit to a Enterprise** | Provides the critical resource life cycles necessary to then specify the design and build sequence for all IT assets. |

| | |
|---|---|
| **Row (2,3)** | Business |
| **Column** | Business Information Systems |
| **Metadata work product description** | The non-redundant allocation of business information systems and their descriptions to the high level business functions via the invoking business events.<br><br>This is supplemented by the business event hierarchies and the allocation of those business events to the detailed business function scenarios. |
| **Benefit to a Enterprise** | Known is the next layer of information systems architecture and the interrelationship between the information systems and the resource life cycle nodes. This supports the knowledge that all the database objects and information systems are allocated to and support resource life cycle nodes. |

| | |
|---|---|
| **Row (2,4)** | Business |
| **Column** | Business Events |
| **Metadata work product description** | Event sequencing and their hierarchies.<br><br>A model of the business cycles comprised of an initiating event within calendars and/or a business cycle.<br><br>Business scenarios sequencing and hierarchies. |
| **Benefit to a Enterprise** | The master schedule of business events within cycles and calendars enables the start of the sizing of the various data transfers and the requirements for processing resources.<br><br>Each event within the schedule is then able to identify the timing and sequencing of the business information systems that must operate to acquire and manipulate the data in service to the functions within the organizations that accomplish various missions |

| | |
|---|---|
| **Row (2,5)** | Business |
| **Column** | Business Function |
| **Metadata work product description** | Business scenarios including their sequencing and hierarchies. Each business function hierarchy is set down along with the sequencing of the steps within each hierarchy. If different business organizations perform the business function then the scenario descriptions can be different so long as the ultimate objectives of the function are clearly identified and are obvious to those who perform the function. There can be multiple equivalent functional hierarchies because one may be supported by automation and the other might not. Also, one might be styled differently to reflect decentralization or centralization. Regardless the totality of the functions should be represented. |
| **Benefit to a Enterprise** | The enterprise is able to contain multiple functional based styles to accomplishing human-based work independently from the computer support that may be provided. The functions are linked to the organizations and to business events as appropriate. |

| | |
|---|---|
| **Row (2,6)** | Business |
| **Column** | Organizations |
| **Metadata work product description** | The model of the actual enterprise allocation of responsibilities and specification of work products. Typically, an organization chart expresses the allocation of responsibilities, but other supporting documents describe the work products. To be complete, the organization chart would have to be supplemented with work products (e.g., control work, coordination work, and operational work) and the originating and receiving organization units identified.<br><br>Interrelationships between organizations and missions, and then between mission-organizations and functions.<br><br>Allocation of key organizational responsibilities to the mission-organization-functions.<br><br>Defined as well are the information needs by mission-organization-function including the information need characteristics. These are interrelated to the resource life cycle nodes. |
| **Benefit to a Enterprise** | The benefit to an enterprise is the interrelationships among missions, functions, and organizations at the appropriate level of detail to ensure that all the necessary organizations, functions and missions are properly identified and cross referenced.<br><br>The various information needs are characterized and described and then allocated to the interrelated sets of mission-organization-function so that the various outputs are determined.<br><br>The determined information needs are then allocated to the resource life cycle nodes to then serve as a defendable set of data requirements.<br><br>This enables a full exposition of the information needs as required by resource life cycle nodes in fulfillment of a particular mission-organization-function.<br><br>The information needs are then cross referenced to the Resource Life Cycle nodes to ensure that the database objects that represent and complete. |

| | |
|---|---|
| **Row (3,1)** | System |
| **Column** | Mission |
| **Metadata work product description** | Policy hierarchies that must be followed to ensure a well ordered enterprise. The policies flow directly from the missions as they are the "rules" through which the missions are accomplished.<br><br>Relationships should exist between policy nodes and database objects as database objects are the mechanism for enforcing policy and for squarely representing the proof that policy is enforced. |
| **Benefit to a Enterprise** | Identifies the various policies that must be created and/modified and in all cases followed to ensure the correct and timely execution. |

| | |
|---|---|
| **Row (3,2)** | System |
| **Column** | Database Objects |
| **Metadata work product description** | Database objects are distilled from the set of all "nouns" that exist in the database domains. Database domains are "noun intensive" descriptions of the "data" required by each lowest level mission description.<br><br>Determined as well from the database domains are those "nouns" that then represent classes of properties and data elements.<br><br>The identified database objects are cross referenced to the resource life cycle nodes to ensure completeness.<br><br>Based on an analysis of database domains, the database object classes are identified. From these, commonly employed entities, attributes and relationships are created within specified data models.<br><br>Determined also are commonly defined and employed data elements (via ISO 11179 et al) and their semantics.  Identified, created, and allocated to data elements are all value domains. Employ the ability to interrelated data element value domain instances to then manage reference table evolution and value transformations. |

| | |
|---|---|
| **Benefit to a Enterprise** | This represents the complete data-object class specification that ultimately must be reflected in the detailed design of DBMS schema data integrity constraints, rules, stored procedures, or coded into business information systems.<br><br>Data elements represent the non-redundant set of business fact templates that are employed to ensure that where ever used within database designs and however used within business information systems that there is one common definition, semantics, and value domains. |

| | |
|---|---|
| **Row (3,3)** | System |
| **Column** | Business Information Systems |
| **Metadata work product description** | The set of business information system designs that represent the IT support down to the names of the applications. The model could include the controls and mechanisms, as well as the inputs and outputs to the logical systems representations of the system functions/processes. |
| **Benefit to a Enterprise** | The project environment will then have, as required by both its missions and data, the necessary and sufficient set of business information system designs that are able to be cross referenced to all mission and data requirements.<br><br>The required state transitions relate to the set of resource life cycle nodes and the interactions between the nodes of one resource and those of another resource.<br><br>Once known, the exact sequence for database and business information system builds can be determined. |

| | |
|---|---|
| **Row (3,4)** | System |
| **Column** | Business Events |
| **Metadata work product description** | Business event invocation protocols, input and output data and messages.<br><br>Relationship with business function use of best practices and business information systems. |
| **Benefit to a Enterprise** | Identifies the sequencing of the information systems processing requirements both from within the domain of a project and from within the domain of the feeder organizations. |

| | |
|---|---|
| **Row (3,5)** | System |
| **Column** | Business Function |
| **Metadata work product description** | Best practices, quality measures and accomplishment assessments. These materials represent the idealized methods an organization can employ to accomplish business functions. Supporting each best practice are the various performance targets and assessments that judge satisfactory accomplishment. Whenever business functions are performed differently there must be style independent assessments. |
| **Benefit to a Enterprise** | These serve as guides to the development of the functions that are to be performed by the various organizations. As each requires an assist from automation the appropriate business event within a business cycle or calendar is located or created and linked to this particular aspect of a business function. |

| | |
|---|---|
| **Row (3,6)** | System |
| **Column** | Organizations |
| **Metadata work product description** | Best practices, quality measures and accomplishment assessments.<br><br>Relationship between job roles, and also with business event invocation. |
| **Benefit to a Enterprise** | Enumerates the activities that must be accomplished to achieve the goals and objectives of the project from within the identified business functions of the various organizations. Identifies the human resource requirements on contributing staff within the participating organizations. |

| | |
|---|---|
| **Row (4,1)** | Technology |
| **Column** | Mission |
| **Metadata work product description** | Policy execution enforcement mechanisms. The various business rules that surface during database design, database object design, business information system design, business event, and business function design are detailed as appropriate to their implementation paradigm and tested. Collectively they are examined to determine conflicts and the conflicts resolved as may be appropriate |
| **Benefit to a Enterprise** | These enable project management to know what must be accomplished by whom and when to have a well ordered systems environment. |

| | |
|---|---|
| **Row (4,2)** | Technology |
| **Column** | Database Objects |
| **Metadata work product description** | Database object class specifications are completed though the creation of tables, columns, embedded processes, states and state transformations. These are then represented in the various implemented data models required for each of the database architecture classes. Relationships are created to all attributes and data elements. Refined value domains are created and allocated as necessary. |
| **Benefit to a Enterprise** | These represent the actual logical databases and the required processes and states that must be represented in each of the databases of the various data architecture classes.<br><br>Because all this is interrelated within the same metadata repository a complete set of "proofs" can be produced to then ensure that all stakeholder needs are represented. |

| | |
|---|---|
| **Row (4,3)** | Technology |
| **Column** | Business Information Systems |
| **Metadata work product description** | Completed detailed business information system designs. Included as well are all the client/server and/or batch execution mechanisms that must exist. |
| **Benefit to a Enterprise** | Similar to the data architecture, these business information system designs then represent the data acquisition, storage, extraction, and manipulation activities that are supported. Again, because the business information system artifacts exist within an integrated metabase then the business event cycle and calendar executions of these business information systems can be produced. Produced as well is the complete cross referencing between the business information system and the implemented database designs. |

| | |
|---|---|
| **Row (4,4)** | Technology |
| **Column** | Business Events |
| **Metadata work product description** | Presentation layer information system instigators. Determined are the actual execution sequence for the various business information systems as they support the accomplishment of business functions. |
| **Benefit to a Enterprise** | This determines the systems environment sequencing and timing of data acquisition, storage, manipulation, and reporting. |

| | |
|---|---|
| **Row (4,5)** | Technology |
| **Column** | Business Function |
| **Metadata work product description** | Activity sequences to accomplish business scenarios. |
| **Benefit to a Enterprise** | Business functions are detailed into their specific activity sequences that accomplish the business scenarios. Each set of activities are stylized to fit the specific organization carrying them out. The activity sequences are evaluated against the best practices and assessment criteria to ensure that the activities accomplish the desired result.<br><br>As each requires an assist from automation the appropriate business event within a business cycle or calendar is located or created and linked to this particular aspect of a business function. |

| | |
|---|---|
| **Row (4,6)** | Technology Model |
| **Column** | Organization |
| **Metadata work product description** | Procedure manuals, task lists, quality measures and assessments. |
| **Benefit to a Enterprise** | The information produced supports the determination of the quantities, content, distribution and training of the necessary procedure manuals, task lists, quality measures and assessments, and the quantity of staff and impact on their current work environment required by the organization to accomplish their business functions in support of enterprise missions. |

| | |
|---|---|
| **Row (5,1)** | Deployment |
| **Column** | Mission |
| **Metadata work product description** | Installed business policy and procedures. These are detailed to the level of individual business units so that they may have day to day guidance on the accomplishment of work. |
| **Benefit to a Enterprise** | This provides a project the mechanisms to know when and how well the various operations are taking place. |

| | |
|---|---|
| **Row (5,2)** | Deployment |
| **Column** | Database Objects |
| **Metadata work product description** | Operational data model and view data models are created to support access through the implemented business information systems.<br><br>View models to store the interrelationships among view columns and relate the view column intersections to support ETL process specifications. |
| **Benefit to a Enterprise** | The operational data model is a physical database that is required to support the computing environment, the DBMS' capabilities, and the quantity of data to be supported.<br><br>The operational (physical) data models are cross referenced to the implemented (logical) data models. This then supports the environment wherein multiple and different physical database deigns are required .<br><br>In addition to physical database designs, the view models are created so that if there are a local set of names to be supported then they can. Additionally the view models are needed for server supported data transformations, complex data accesses, and likely row-level security.<br><br>Because of the view models, then regardless of the database accessing language, the required data integrity processes will be executed.<br><br>Because of the integrated metadata environment, evolution and maintenance of databases is greatly eased. |

| | |
|---|---|
| **Row (5,3)** | Deployment |
| **Column** | Business Information Systems |
| **Metadata work product description** | Implemented business information systems throughout the environment and across the different layers of technology. Supporting the business information systems are hotline, technical support, user guides, training courses, and on-line assists from Intranet websites. |
| **Benefit to a Enterprise** | These represent the business information systems that are to be operating within the environment.<br><br>Because of the integrated metadata environment, evolution and maintenance of business information systems is greatly eased. |

| | |
|---|---|
| **Row (5,4)** | Deployment |
| **Column** | Business Events |
| **Metadata work product description** | Client and server windows and/or batch execution mechanisms. |
| **Benefit to a Enterprise** | This provides presentation layer execution mechanisms related to business information systems interface with business function policies, procedures, in need of automation support. |

| | |
|---|---|
| **Row (5,5)** | Deployment |
| **Column** | Business Function |
| **Metadata work product description** | The actual office procedures employed by organizations performing business functions. These deployed activities must be supported by necessary operational policies and procedures and what ever technology supports that many be required. |
| **Benefit to a Enterprise** | surfaces the actual policies and procedures that are to be accomplished within various business cycles and events, and within various organizations. Enables simulations of business activities before they are set into actual execution. |

| | |
|---|---|
| **Row (5,6)** | Deployment |
| **Column** | Organizations |
| **Metadata work product description** | Office policies and procedures to accomplish activities.<br><br>Daily schedules, shift and personnel assignments. |
| **Benefit to a Enterprise** | This provides detailed looks at how various user groups are organized and are working together across agreed-upon business cycles and calendars to contribute data and to make use of data for analyses and reports as soon as it becomes available. |

| | |
|---|---|
| **Row (6,1)** | Operations |
| **Column** | Mission |
| **Metadata work product description** | The ongoing and executing set of policies that carry out various aspects of the enterprise' missions. |
| **Benefit to a Enterprise** | As missions are accomplished, statistics about their accomplishment can be captured and analyzed for activity reporting, length of time for execution, and for feedback for a next round of data and process improvements. |

| | |
|---|---|
| **Row (6,2)** | Operations |
| **Column** | Database Objects |
| **Metadata work product description** | The view models that represent the interrelationship between the databases and the business information systems. |
| **Benefit to a Enterprise** | During the execution of business information systems, statistics about the databases such as size, row quantities by table, volumes, and velocities can be collected and stored in support of operational data and process statistics. |

| | |
|---|---|
| **Row (6,3)** | Operations |
| **Column** | Business Information System |
| **Metadata work product description** | Operating business information systems |
| **Benefit to a Enterprise** | During the execution of business information systems, the volume and velocity of business transactions can be collected across all the different database architecture classes, user communities, and types of business transactions. |

| | |
|---|---|
| **Row (6,4)** | Operations |
| **Column** | Business Event |
| **Metadata work product description** | Start, Stop, and messages |
| **Benefit to a Enterprise** | This provides visibility into the day to day operational aspects of interfacing business functions and business information systems. This involves ensuring that there are enough data entry forms, sufficient paper for reports, computers, and telecommunications networks. |

| | |
|---|---|
| **Row (6,5)** | Operations |
| **Column** | Business Function |
| **Metadata work product description** | Detailed procedure based instructions. |
| **Benefit to a Enterprise** | These provide the detailed instructions that exist within an office and a schedule to actually perform the business function's work. These office procedures should be taught, monitored, and constantly evaluated for maximum efficiency, effectiveness, and minimum cost and risk. |

| | |
|---|---|
| **Row (6,6)** | Operations |
| **Column** | Business Organization |
| **Metadata work product description** | Daily activity executions and assessments. |
| **Benefit to a Enterprise** | Organizations are able to be individually styled with their own way of accomplishing business functions so long as they operate within generally accepted business calendars and cycles. As organizations perform their work within these styled functions, statistics regarding transaction frequencies and volumes can be gathered in support of measuring different organizations, different management styles, centralization or decentralization, and the like. |

## 5.0 Metadata Management System

From the previous sections, the mission of a comprehensive metadata repository is to provide metadata in response to at least the following needs:

- *The essential missions that define the very existence of the enterprise, and that are the ultimate goals and objectives that measure enterprise accomplishment from within different business functions and organizations.*

- *The procedures performed by groups in their achievement the various missions of the enterprise from within different enterprise organizations.*

- *The organizations that are accomplishing what aspects of missions with what databases, information systems and through which functions.*

- *The key Resources (facilities, materiel, staff, etc.) How are they sequenced, interrelated, and how are they supported through databases and information systems.*

- *The information (a.k.a. query results or reports) is needed by various organizations in their functional accomplishment of missions and what databases and information systems provide this information.*

- *The data is needed by functional proponents, how is it defined within data architectures and databases and how and where are those databases deployed and then used by business information systems in support of mission accomplishment.*

- *The context independent semantic templates of data elements and how are these configured into models of data (the consequence of policy execution) determined as needed by functional experts in support of enterprise missions, and how are these specified data model requirements configured into implemented databases that ultimately operate within various organizations as they perform the functions needed by enterprise missions.*

- *The business information systems, where are they, how are they related to mission, organization, function, and databases. What is the impact on these business information systems when policy (a.k.a., data) is required or changed.*

- *The identification, allocation, and scheduling of database and information system projects within the context of Resource Life Cycles. Resulting project schedules are then able to be accomplished in a business resource defined sequence to best achieve enterprise missions through business functions and organizations.*

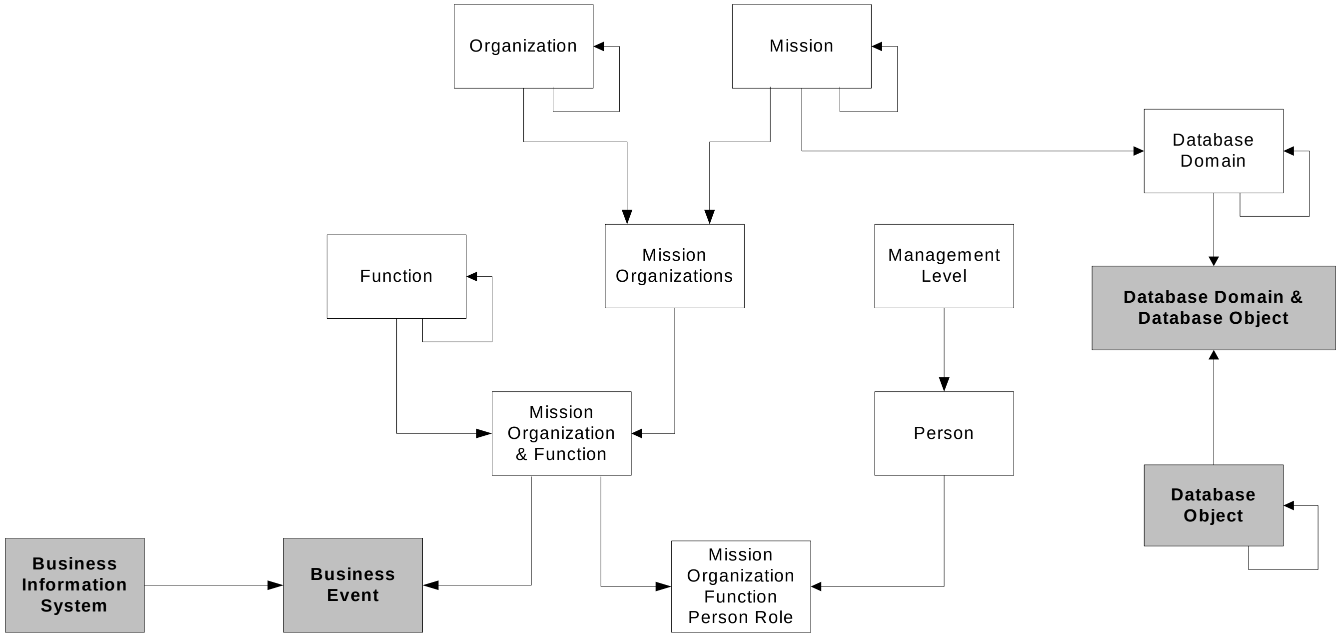
- *The identification, estimation, and then monitoring of database and business information system projects from within the context of well established metrics, and templates work breakdown structures, and deliverables.*

## 5.1 Data Architecture

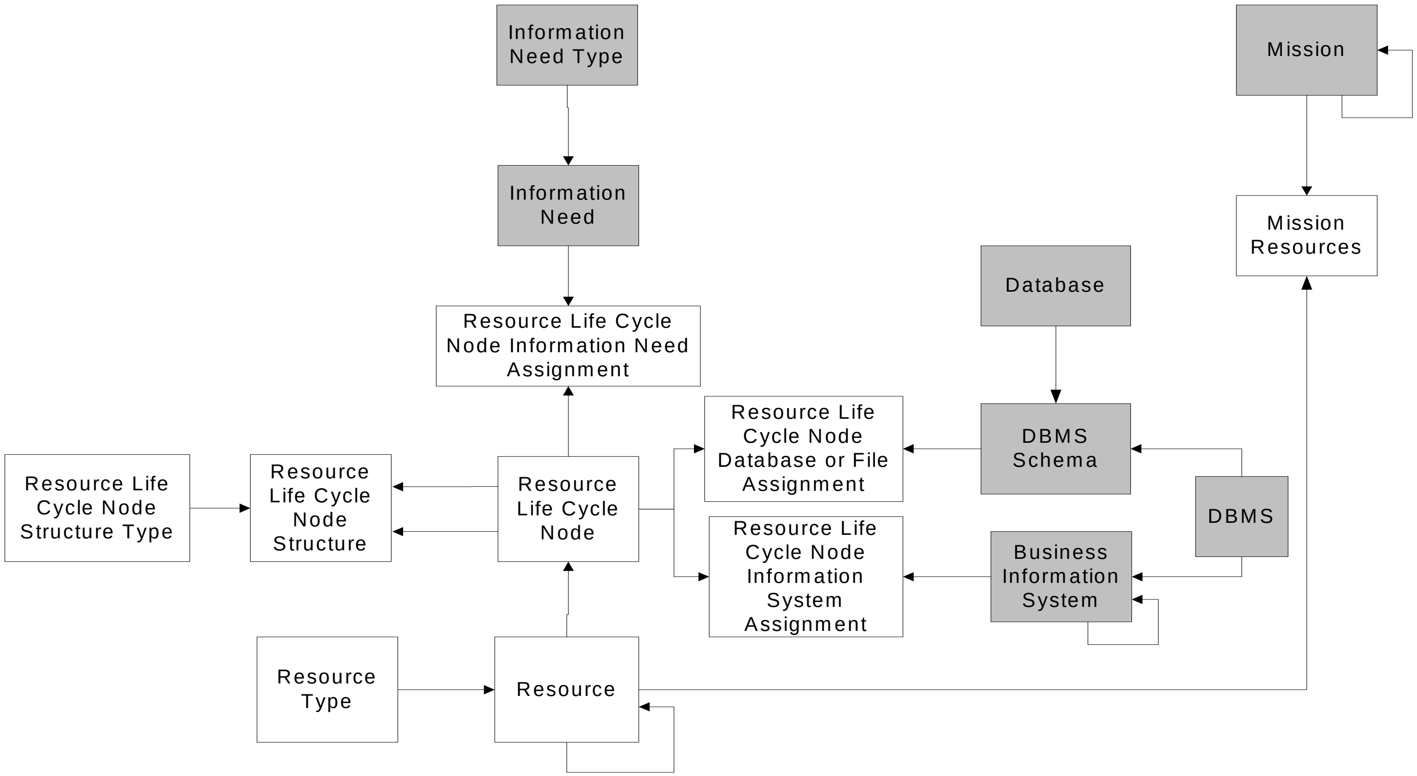
It is not sufficient to merely infer and then list all the infrastructure work products that must be produced and managed, or to use a collection of CASE and modeling tools such as Erwin that do not derive their data from a single metadata repository. If it were sufficient, then the traditional set of project documents, presentations, and "ERwin" data models would have be adequate to make the content of all these materials truly useful, their content must be cast into database tables and completely interrelated into an the metadata repository. The following high level metadata object classes are necessary:

Application View Data Models
Business Calendars
Business Events
Business Information Systems
Conceptual Data Models
Database Domains
Database Object Classes
Databases
Functions
Information Needs and Characterizations
ISO 11179 Data Element Metadata
Logical Data Models
Missions
Organizations
Persons and Roles
Physical Data Models
Projects Management
Resources and Life Cycles

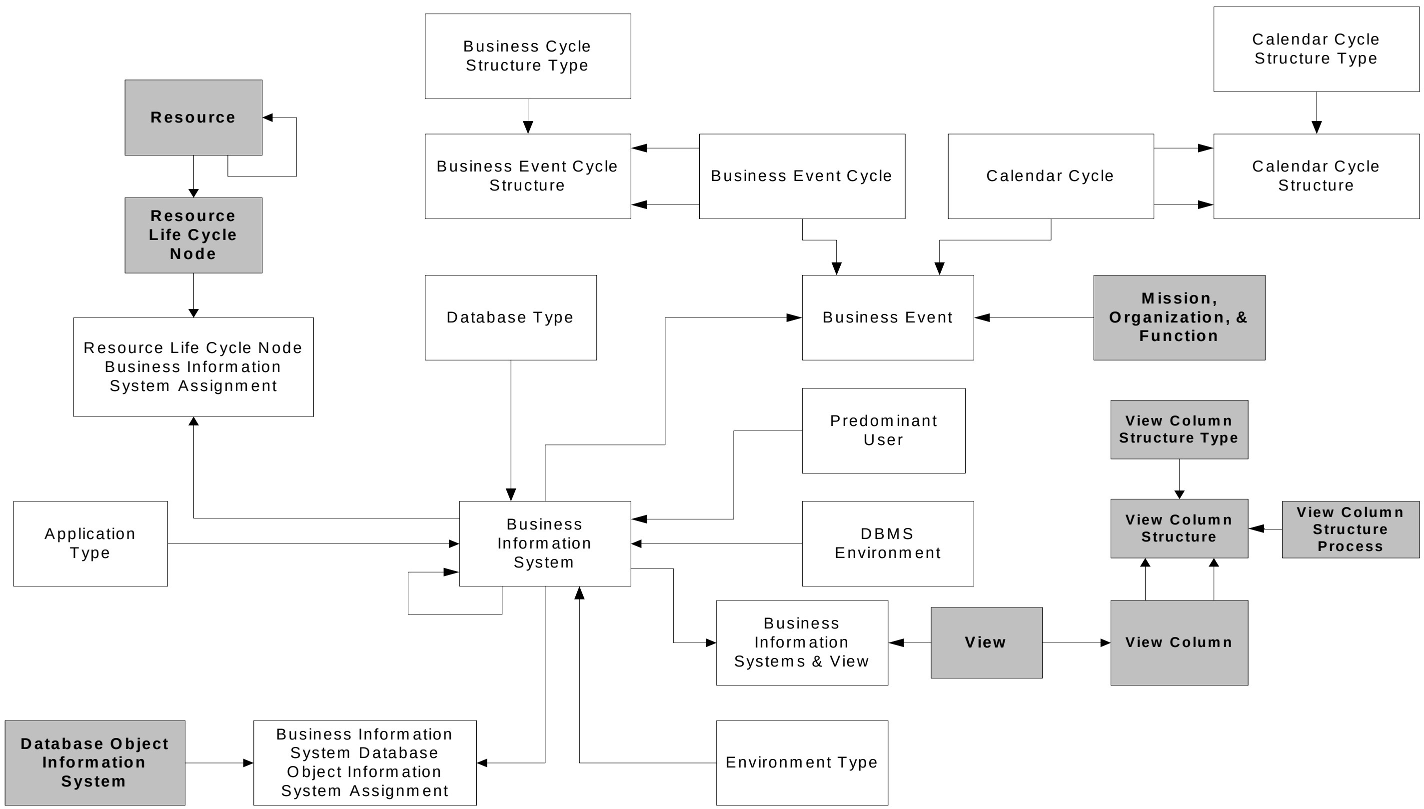
The diagrams that follow are illustrative of a set of entity relationship diagrams of the work products except for the Technology Architecture column. Each "box" represents a database table, and the lines among the boxes represent database relationships. A metadata repository system is the database application built around these data models. At Whitemarsh, the metabase is that repository. When the metadata repository is then employed, the project infrastructure work products that result from analysis, design, implementation, and maintenance efforts will be comprehensively managed. In the diagrams on the pages that follow, those that are shadowed (i.e., gray) are updated in a different repository module than would be inferred from the diagram. At the end of the diagrams are the metabase's meta entity definitions.



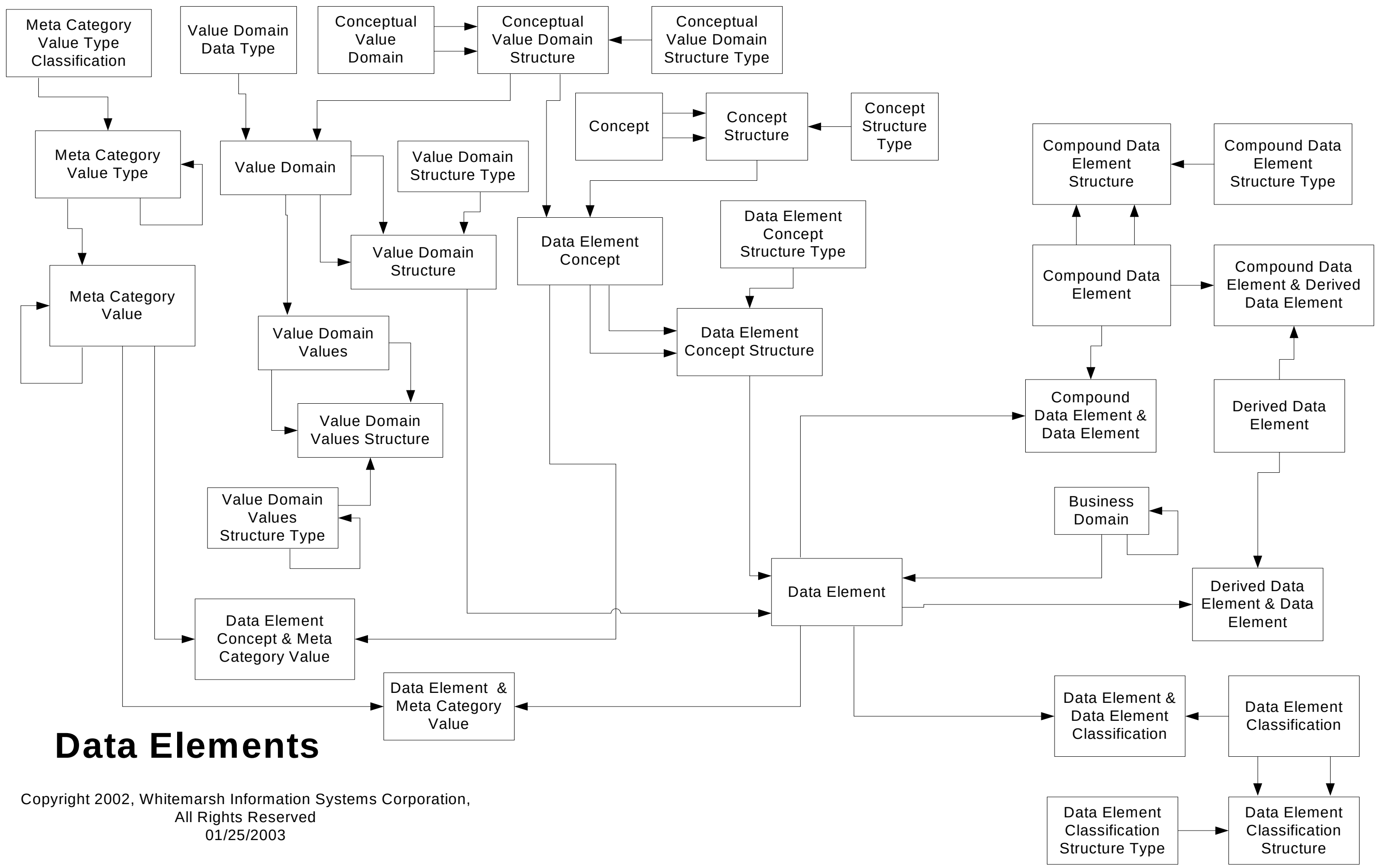

# Mission, Organization, Function, Person Analysis



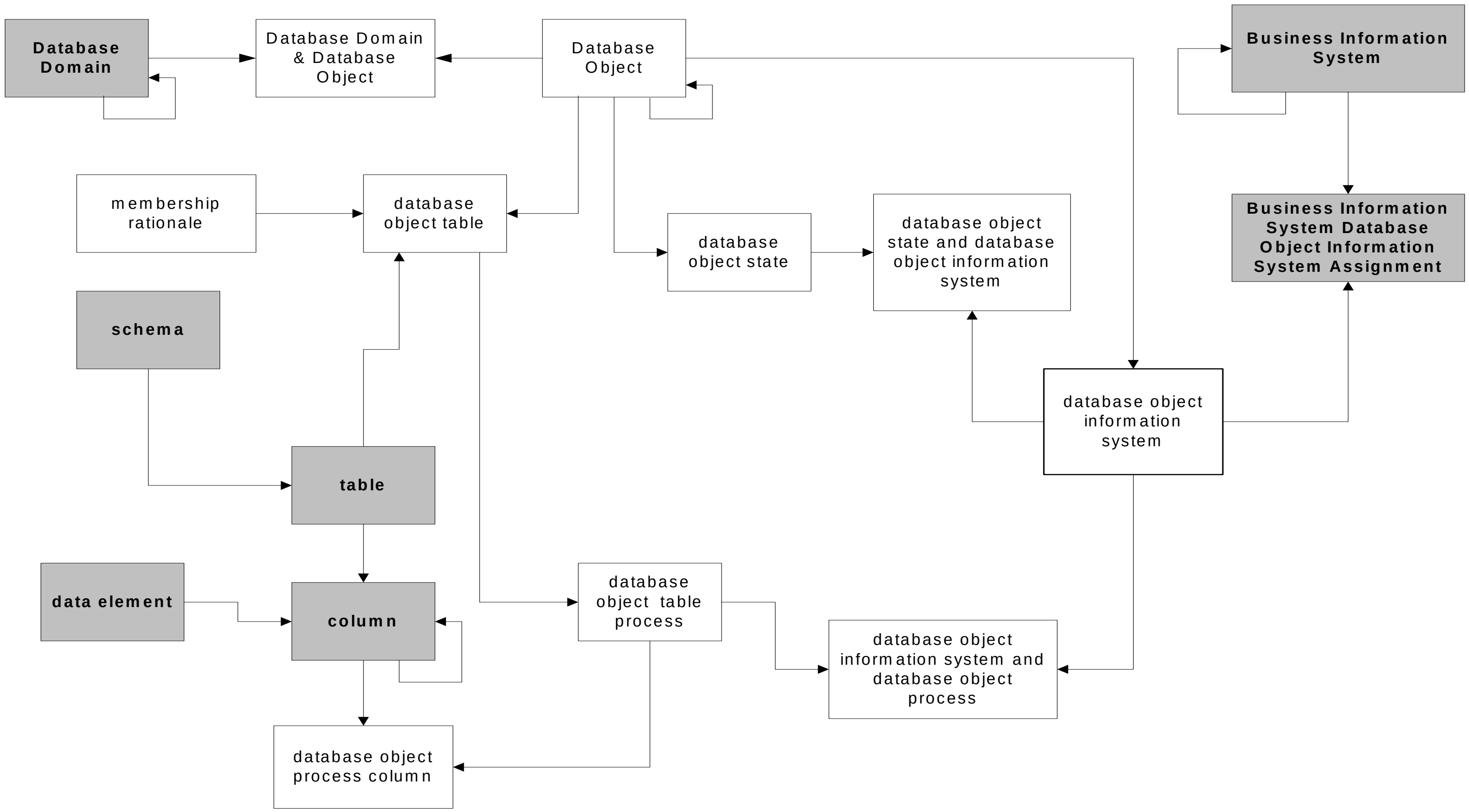

**Resource Life Cycle Analysis**



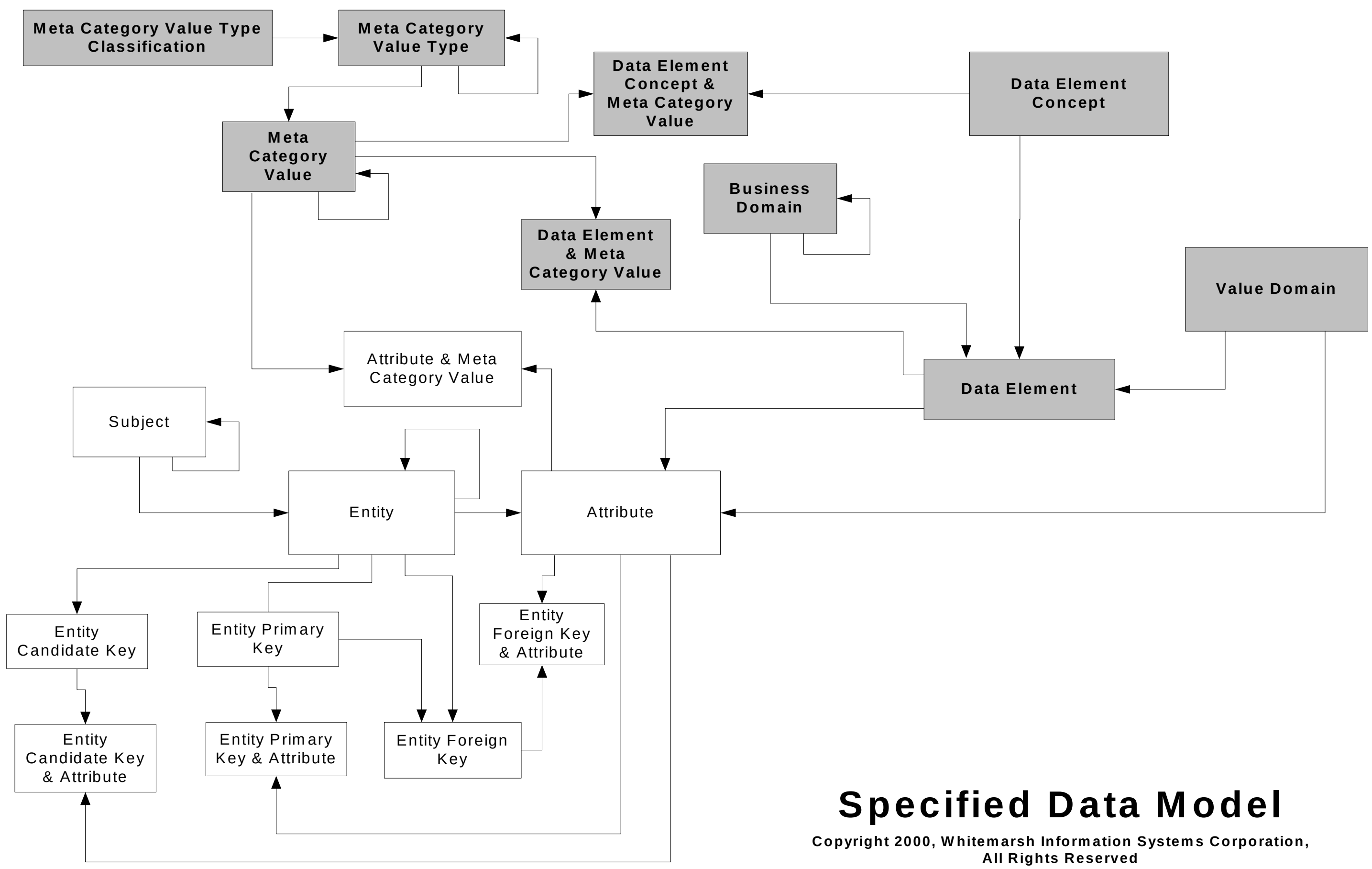

**Business Information Systems**




## Data Elements

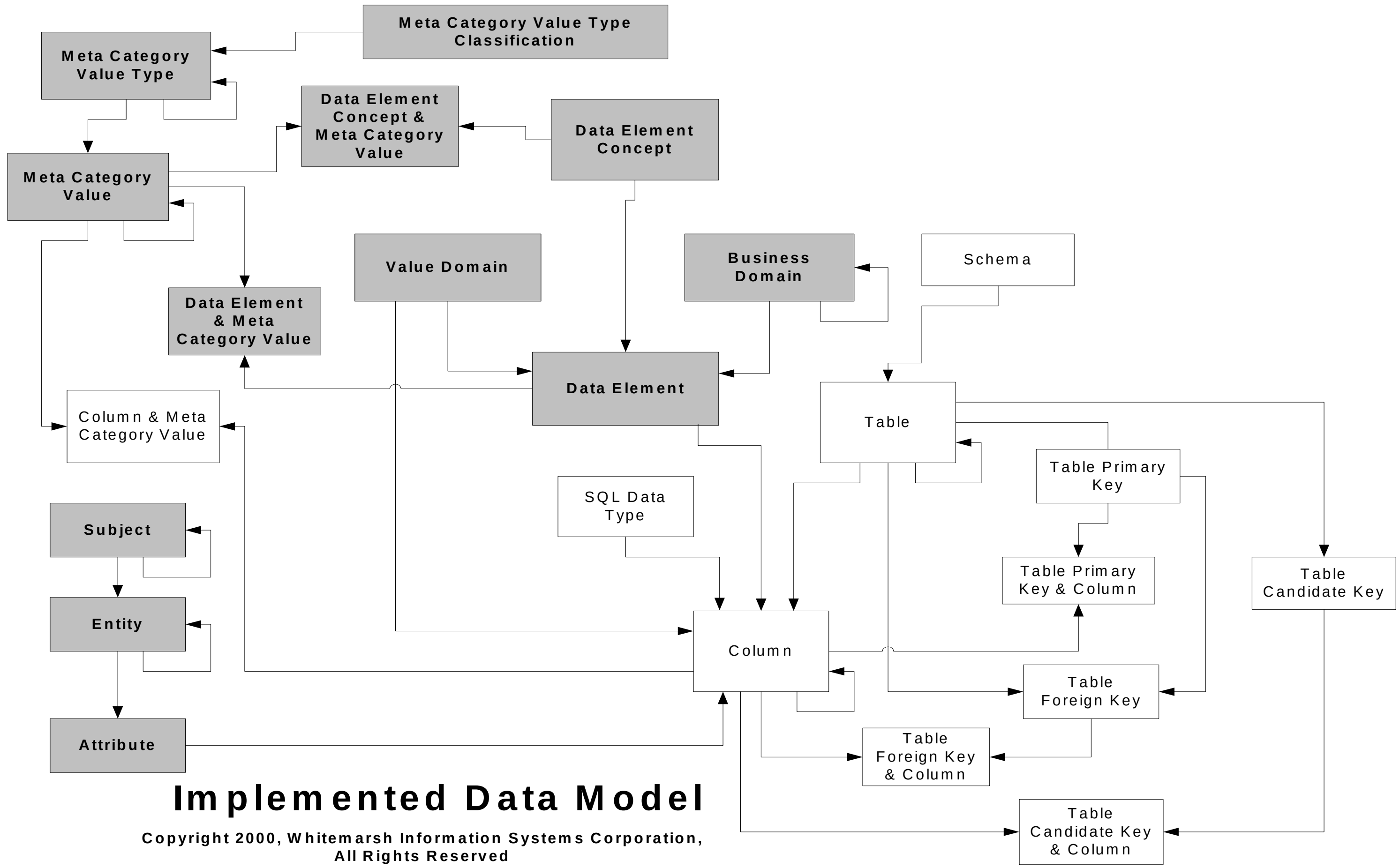



Database Objects



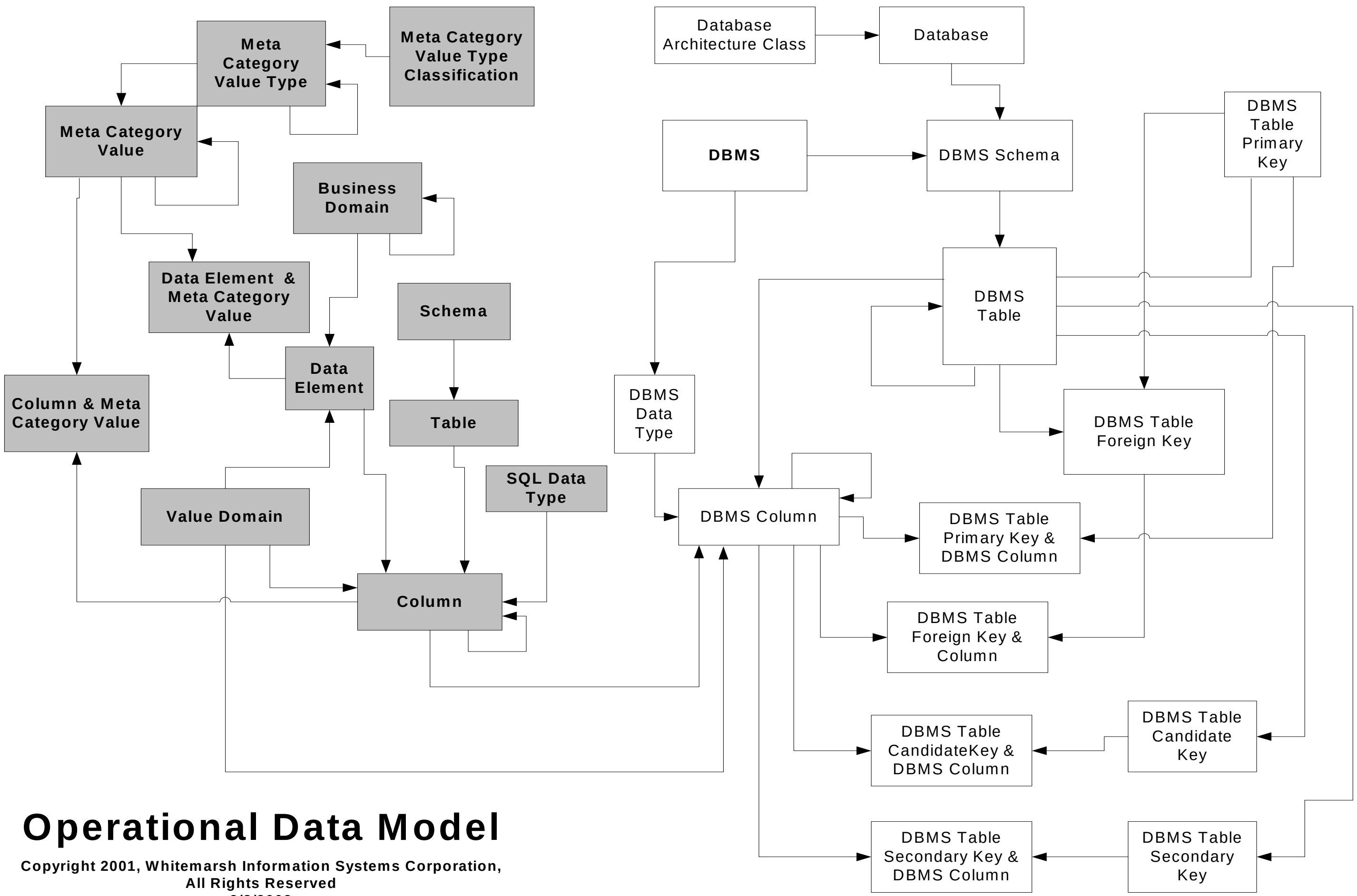

## Specified Data Model


3/08/2002



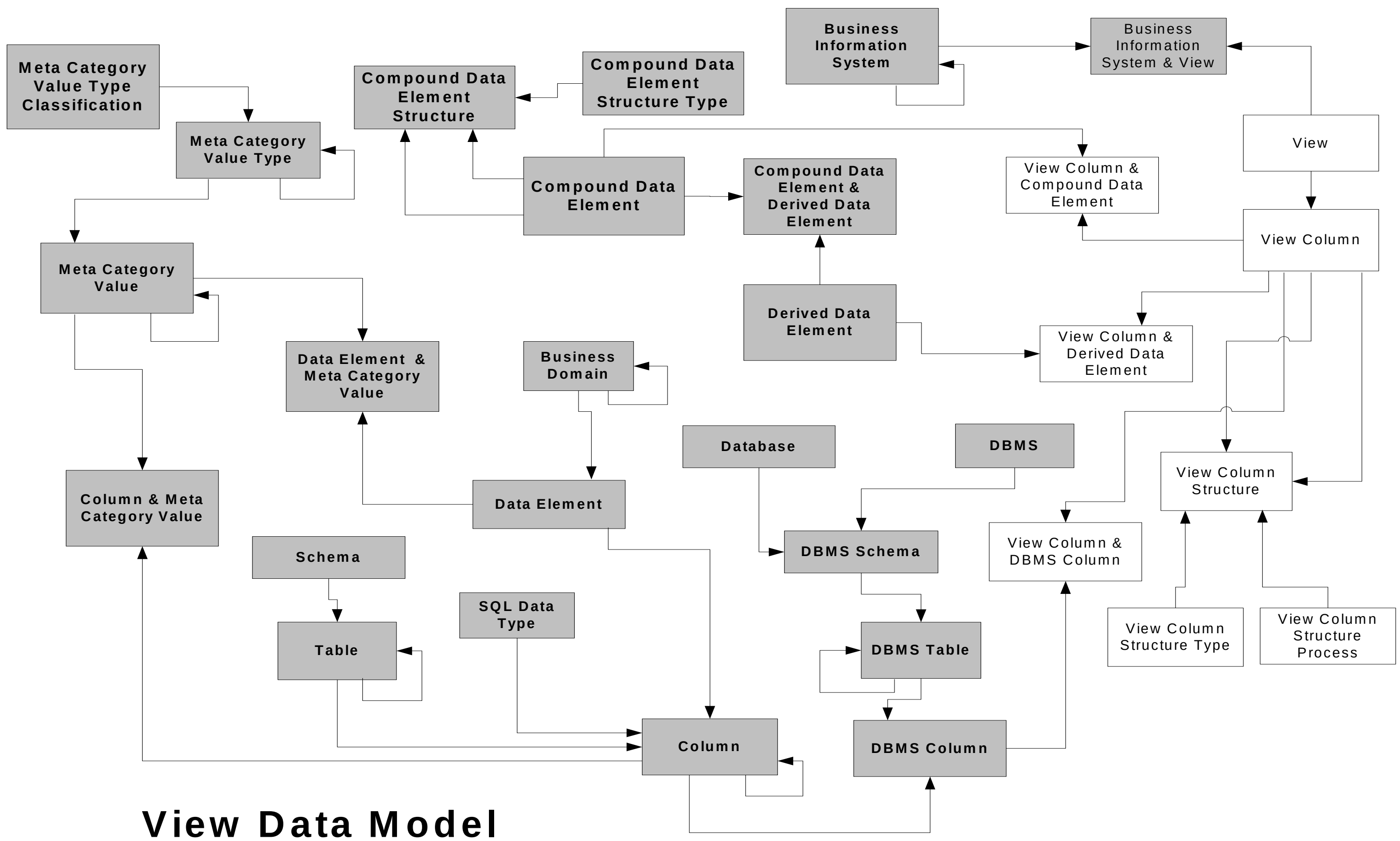

# Implemented Data Model


3/8/2002

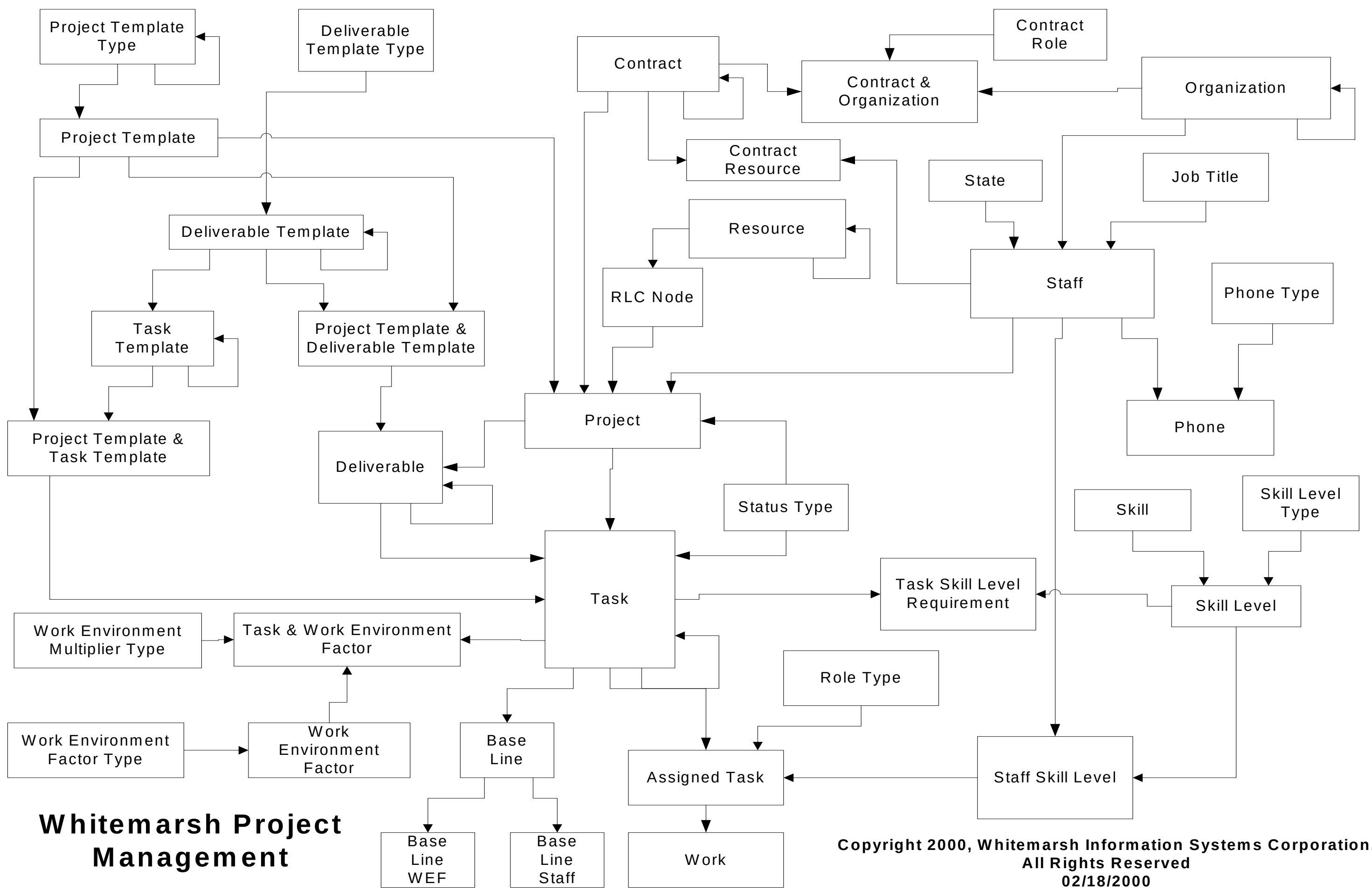



## Operational Data Model


3/8/2002




**View Data Model**


Project Template Type
Deliverable Template Type
Contract
Contract Role
Contract & Organization
Organization
Project Template
Contract Resource
State
Job Title
Deliverable Template
Resource
RLC Node
Staff
Phone Type
Task Template
Project Template & Deliverable Template
Project Template & Task Template
Deliverable
Project
Phone
Status Type
Skill
Skill Level Type
Task
Task Skill Level Requirement
Skill Level
Work Environment Multiplier Type
Task & Work Environment Factor
Role Type
Work Environment Factor Type
Work Environment Factor
Base Line
Assigned Task
Staff Skill Level
Whitemarsh Project Management
Base Line WEF
Base Line Staff
Work
Copyright 2000, Whitemarsh Information Systems Corporation, All Rights Reserved
02/18/2000

## Metabase Entity Definitions

**Application Type.** An Application Type is a classification of the application such as distribution, finance, human resources and the like.

**Attribute & Meta Category Value**. Attribute & Meta Category Values are the relationship that exists between an attribute and its set of assigned meta category value semantics. These assigned semantics are always a subset of those assigned to the attributes “parent” data element through processes within the data modeler.

**Attribute Value Domain.** Attribute value domain are collections of a more refined set of value domains for specific attributes. Attribute value domains are a subset of data element value domains.

**Attribute.** Attributes are the manifestation of the semantics of a data element within an entity of a subject. Attributes may have additional semantics that further refine it within th e context of the data element.

**Business Domains.** Business Domains are the parent of one or more data elements and enable data elements that are named the same to be distinguished by the business domain in which they reside.

**Business Event Cycle Structure**. A Business Event Cycle Structure is a collection of business event cycles, for example, a Summer cycle may also consist of a End of School cycle, Back to School Cycle, Vacation Cycle, and a Holiday Cycle.

**Business Event Cycle Structure Type**. A Business Event Cycle Structure Type classifies a collection of Business Event Structure instances.

**Business Event Cycle.** A Business Event Cycle is a cycle during which business events occur such as financial reports, holidays, business planning and the like. A business event cycle may be simple or complex. If complex then the business event cycle actually consists of other business event cycles as represented in the business event cycle structure.

**Business Information System Database Object Information System Assignment.** A Business Information System Database Object Information System Assignment records the database object information systems that are invoked by the business information system.

**Calendar Cycle.** A Calendar Cycle is a set of recurring calendar based dates that are of interest to the enterprise. For example, quarterly, bi-weekly, monthly, daily, and the like. Calender cycles are linked to Business Events so that the timing of business event triggering can be know.

**Business Information System**. A Business Information System is a computer based information system that is being managed through the metabase. It is know by its characteristics, its operation cycles (business event and calendar), subordinate business information systems, employed databases, views, and associated resource life cycle nodes.

**Business Information System and View**. A Business Information System and View is an association between a view and a business information system. This then enables knowledge of the DBMS columns and DBMS tables that are accessed by the business information system.

**Calendar Cycle Structure Type.** A Calendar Cycle Structure Type is a classification of a set of Calendar Cycle Structures.

**Calendar Cycle Structure**. A Calendar Cycle Structure is a collection of calendar cycles, for example, a Financial Report cycle may consist of a Second Week of the Month, the Last Friday of the month, and the first day of the quarter.

**Characteristic Type**. A characteristic type is a collection of characteristics that would apply to one or more [information need] characteristic. For example, cut-off, media of production, distribution, and the like.

**Characteristic**. An [information need] characteristic is a collection of characteristics within an information need type that can then be assigned to an information need. For example, if the characteristic type is Media of Production, then the probable set of characteristics could be. hard-copy or on-line access.

**Column & Meta Category Value**. Column & Meta Category Values are the relationship that exists between an column and its set of assigned meta category value semantics. These assigned semantics are always a subset of those assigned to the column's "parent" data element and also a subset of those assigned to the column's "parent" attribute. The order of processing these additional semantics is that the column must first be a subset of the attribute, which in turn must be a subset of the data element.

**Column Value Domain.** Column Value Domains are collections of a more refined set of value domains for specific columns. Column value domains are a subset of attribute value domains.

**Columns**. Columns are the manifestation of the semantics of a data element within a table of a schema. Additionally, a column is a deployment of the semantics of an attribute. Columns may have additional semantics that further refine the column within th e context of either the attribute or the data element. The order of processing these additional semantics is that the column must first be a subset of the attribute, which in turn must be a subset of the data element. Not all the columns of a table must map to attributes from a single entity.

**Compound Data Element & Data Element.** Compound Data Element & Data Element represents the mapping between the one or more data elements and one or more compound data elements

**Compound Data Element Structure Type.** Compound Data Element Structure Type represents the classification of structure among a set of compound data elements.

**Compound Data Element Structure.** Compound Data Element Structures represents a collection of compound data elements that are in turn a compound data element.

**Compound Data Element.** Compound Data Elements consist of multiple data elements in a specific sequence. Compound data elements may also consist of other compound data elements which too appear in a specific sequence

**Concept Structure Type.** Concept Structure Types represents the classification of structure among a set of concepts.

**Concept Structure.** Concept Structures represent a collection of concepts that are related and are in turn represented by a collective concept.

**Concept**. Concepts represent the sets of ideas, abstractions, or things in the real world that are identified with explicit boundaries and meaning and whose properties and behavior follow the same rules. Concepts are used as a basis for specifying the concepts of data elements.

**Conceptual Value Domain Structure Type.** Conceptual Value Domain Structure Type represents the classification of structure among a set of conceptual value domains.

**Conceptual Value Domain Structure.** Conceptual Value Domain Structures represent a collection of conceptual value domains that are related and are in turn represented by a collective conceptual value domain

**Conceptual Value Domain.** Conceptual Value Domains represent the collection of concepts supporting value domains sets from which all value domains. A conceptual value domain forms a basis for a data element concept.

**Data Element & Data Element Concept.** Data Element & Data Element Concept represents the assignment of one or more data elements to one or more data element concepts

**Data Element & Meta Category Value**. Data Element & Meta Category Value represents the assignment of one or more data elements to one or more meta category values.

**Data Element Classification Structure Type.** Data Element Classification Structure Types represent the classification of structure among a set of data element classifications.

**Data Element Classification Structure.** Data Element Classification Structure represent a collection of data element classifications that are related and are in turn represented by a collective data element classification.

**Data Element Classification.** Data Element Classifications are schemes or ontologies to understand the knowledge categories within which data elements reside.

**Data Element Concept Structure Type.** Data Element Concept Structure Type represent the classification of structure among a set of data element concepts.

**Data Element Concept Structure**. Data Element Concept Structure represent a collection of data element concepts that are related and are in turn represented by a collective data element concept.

**Data Element Concept**. Data Element Concepts represent the underlying concepts that support understanding the data element. A data element concept is the association of a concept and conceptual value domain.

**Data Element**. Data Elements are context independent business fact templates. The complete set of semantics for a data element are those explicitly assigned, that is, meta category values, data element concepts, data element classifications, and business domains. It's inherited semantics include those assigned to its containing data element concept and value domains. The value domains of the data element are specifically assigned.

**Data Elements Data Element & Data Element Classification**. Data Elements Data Element & Data Element Classification represents the assignment of one or more data elements to one or more data element classifications.

**Database Architecture Class.** Database Architecture Class is a characterization of a database. That is, original data collection, transaction data staging area (TDSA), subject area database, data warehouse (wholesale or retail), or reference data. An explanation of these database architecture classes is presented in a paper named, Data Architectures, that is on the Whitemarsh website.

**Database Domain & Database Object**. A Database Domain & Database Object is the association of one or more database objects with one or more database domains.

**Database Object Information System**. A Database Object Information System is a collection of processes defined within the domain of the DBMS usually as a stored procedure that transforms one or more rows of a database object from one valid state to another. A database object information system accomplishes one or more database object table processes.

**Database Object Information Systems & Database Object Process.** A Database Object Information Systems & Database Object Process is the association of one or more Database Object Information Systems and Database Object Processes. These are set into a sequence for proper accomplishment.

**Database Object State and Database Object Information System.** A Database Object State and Database Object Information System is the association of one or more Database Object States and Database Object Information Systems. The association exists within a sequence so that the state is properly achieved.

**Database Object State.** A Database Object State is a well defined value state of a database object. States occur in a particular sequence, typically from the null state through a set of value states and returning to a null state. A database object state is accomplished through one or more database object information systems.

**Database Object Table Process Column.** A Database Object Table Process Column is an association of a specific database object table process and a specific column of a table.

**Database Object Table Process.** A Database Object Table Process is a process such as insert, change, or delete that occurs against one row of a single table within a database object. A table owns (and is thus acted upon by) one or more database object table processes. A database object table process may be invoked by one or more database object information systems.

**Database Object Table**. A Database Object Table is an association of a table with a database object. Membership rationale classifies the reason why a table belongs to the database object.

**Database Object**. A database object is a collection of one or more tables. The rows from these tables are transformed from one valid state to another via database object table processes and database object information systems. Database objects are related to one or more database domains.

**Database Type.** A Database Type is a classification of a style of database design. This is also called data architecture class and will be replaced with that meta-entity. The most common data architecture classes are original data capture, transaction data staging area, reference data, wholesale data warehouses and retail data warehouses (also called data marts). A list and brief definition of each of the database architecture classes is contained in Section 4, Reference Data.

**Database.** A Database is named collection of DBMS tables within the structure of a DBMS schema. A well designed database presents a unified set of policy evidenced through the rows of data within a specific subject area.

**DBMS Column**. DBMS Columns are the manifestation of the semantics of a column within a DBMS table of a DBMS schema. Not all the DBMS columns of a DBMS table must map to attributes from a single table.

**DBMS Data Type.** A DBMS Data Type is a DBMS vendor's classification of the values represented by a column of a row of data. The data types common represented are character, integer, binary, and the like. Each DBMS data type imposes a set of rules regarding allowable values and allowed operations on the values. For example adding an integer value to a date value, but disallowing the adding of two dates.

**DBMS Environment**. The DBMS Environment meta entity is intended to carry information that would indicate that the business information system is serviced by one or more than one DBMS such as Oracle or Sybase.

**DBMS Schema.** DBMS Schema. represent a database structure of DBMS tables and relationships within the enterprise. Operational data models data models are cast within the domain of a DBMS schema. The set of all DBMS tables within a DBMS schema is not required to be taken from a single set of entities within a subject area.

**DBMS Table Candidate Key & DBMS Column**. DBMS table candidate key & DBMS columns are the relationship between an DBMS table candidate key and the DBMS columns that comprise the key. The DBMS columns exist within a implemented sequence. Candidate key DBMS columns are not allowed to include any DBMS columns within the DBMS table's primary key.

**DBMS Table Candidate Key**. DBMS Table Candidate Keys represent a collection of DBMS columns within an DBMS table that when their values are collectively employed would result in the retrieval or update of a single row of data for that DBMS table. There may be multiple candidate keys within an DBMS table. DBMS columns of candidate keys are not allowed to overlap each other or the DBMS table's primary key.

**DBMS Table Foreign Key & DBMS Column.** DBMS Table Foreign Key & DBMS Columns are the relationship between an DBMS table foreign key and the DBMS columns that comprise the key. The DBMS columns exist within a implemented sequence. Foreign key DBMS columns are not allowed to include any DBMS columns within the DBMS table's primary key.

**DBMS Table Foreign Key**. DBMS Table Foreign Key. represent a related DBMS table's primary key. The name of the foreign key should match closely the relationship that the key is to represent. The DBMS columns of the foreign key should be able to be deleted entirely from the DBMS table without any loss of policy. The DBMS columns of the foreign key are not allowed to overlap the DBMS columns of the DBMS table's primary key. In addition to the foreign key's DBMS columns there are additional rules governing inserts, updates, and deletes.

**DBMS Table Primary Key & DBMS Column.** DBMS Table Primary Key & DBMS Column are the relationship between an DBMS table primary key and the DBMS columns that comprise the key. The DBMS columns exist within a implemented sequence. Primary key DBMS columns are not allowed to include any DBMS columns within the DBMS table's primary key.

**DBMS Table Primary Key.** DBMS Table Primary Key represent a collection of DBMS columns within an DBMS table that when their values are collectively employed would result in the retrieval or update of a single row of data for that DBMS table if that DBMS table had actually been a DBMS table.
There can only be one primary key within an DBMS table. DBMS columns of primary key are not allowed to overlap each other or the DBMS table's candidate key.

**DBMS Table Secondary Key & DBMS Column**. DBMS Table Secondary Key & DBMS Column are the relationship between an DBMS table secondary key and the DBMS columns that comprise the key. The DBMS columns exist within a implemented sequence. Secondary key DBMS columns are not allowed to include any DBMS columns within the DBMS table's secondary key.

**DBMS Table Secondary Key.** DBMS Table Secondary Key represents a collection of DBMS columns within an DBMS table that when their values are collectively employed would result in the retrieval or update of one or more rows of data for that DBMS. There can be multiple secondary keys within an DBMS table. DBMS columns of secondary key are allowed to overlap each other or the DBMS table's secondary key.

**DBMS Table**. A DBMS Table is intended to be a well defined expression of one policy within a DBMS schema. Ideally, the collection of all the DBMS tables within a DBMS schema area should define a coherent collection policy. A DBMS table may contains DBMS columns that map to columns from multiple tables. This enables operational databases to be non-third-normal form while the implemented data model is. Additionally the DBMS table can have formally defined subtypes.

**DBMS.** DBMS a software system that defines, creates, accesses and maintains databases.

**Derived Data Element & Data Element**. Derived Data Element & Data Element represents the assignment of a data element to one or more derived data elements.

**Derived Data Element.** Derived Data Elements represent the result of some sort of calculation or transformation of either a collection of data elements or compound data elements or a combination of both.

**Entity Candidate Key & Attribute.** Entity Candidate Key & Attributes are the relationship between an entity candidate key and the attributes that comprise the key. The attributes exist within a specified sequence. Candidate key attributes are not allowed to include any attributes within the entity's primary key.

**Entity Candidate Key**. Entity Candidate Keys represent a collection of attributes within an entity that when their values are collectively employed would result in the retrieval or update of a single row of data for that entity if that entity had actually been a table. There may be multiple candidate keys within an entity. Attributes of candidate keys are not allowed to overlap each other or the entity's primary key.

**Entity Foreign Key & Attribute.** Entity Foreign Key & Attributes are the relationship between an entity foreign key and the attributes that comprise the key. The attributes exist within a specified sequence. Foreign key attributes are not allowed to include any attributes within the entity's primary key.

**Entity Foreign Key.** Entity foreign keys represent a related entity's primary key. The name of the foreign key should match closely the relationship that the key is to represent. The attributes of the foreign key should be able to be deleted entirely from the entity without any loss of policy. The attributes of the foreign key are not allowed to overlap the attributes of the entity's primary key. In addition to the foreign key's attributes there are additional rules governing inserts, updates, and deletes.

**Entity Primary Key & Attribute.** Entity Primary Key & Attribute are the relationship between an entity candidate key and the attributes that comprise the key. The attributes exist within a specified sequence. Candidate key attributes are not allowed to include any attributes within the entity's primary key.

**Entity Primary Key**. Entity Primary Keys represent a collection of attributes within an entity that when their values are collectively employed would result in the retrieval or update of a single row of data for that entity if that entity had actually been a table. There can only be one primary key within an entity. Attributes of primary key are not allowed to overlap each other or the entity's candidate key.

**Entity.** An Entity is intended to be a well defined expression of one policy within a subject area. The collection of all the entities within a subject area should define the complete set of policy for that area. Some entities and even some subject areas may never be represented within implemented data models. Additionally, some attributes within an entity may never be employed. Entities may be subtyped.

**Environment Type.** The Environment Type meta entity is intended to distinguish whether the business information system is executing on a desktop, server, or mainframe.

**Information Need Characteristic Assignment**. The information need characteristic assignment is the assignment of the characteristic to the information need. For example, if the information need was a Financial Report, it might be quarterly, hard-copy, and summary.

**Information Need Type**. An information need type is a classification of information need. This may be according to functional area such as finance, administration, sales, etc., or may be some other information need grouping.

**Information Need**. An information need is just what its name implies, a need for information. It should however be formal enough to represent some significant IT assets such as a quarterly financial report, the current state of inventory, an order, and the like. Information needs are created through analysis and are characterized. Information needs are ultimately allocated to Resource Life Cycle Nodes. That is, a database and or an information system is required by a resource life cycle node to produce an information need that in turn fulfills the need of some functional area of an organization that is accomplishing a mission of the enterprise. Information needs are both typed and characterized. While it would not be common, it certainly would be within the scope of an information need to be a data mart that is then processed by OLAP (online analysis processing software).

**Membership Rational**. Membership Rational is a classification of the reasons why a table belongs to a database object as evidenced through the database object table.

**Meta Category Value Type Classification.** Meta Category Value Type Classification are broad classifications of the meta category value types and in turn, meta category values. There are two. prefix and suffix. The prefix meta category value type classification causes all assigned meta category values to appear before the common business name of the data element. Essentially, prefix semantics serve as a collection of “modifiers” to the base semantics of the data element. The suffix meta category value type classification causes all assigned meta category values to appear after the common business name of the data element. Essentially, suffix semantics serve as a collection “class words” that characterize the intended data type, and use of a data element.

**Meta Category Value Type.** Meta Category Value Type are collections of categories within which meta category values reside

**Meta Category Value**. Meta Category Values are the individual semantics that are assigned to data elements in support of the complete specification of the data element. Meta category values are in turn collections within meta category value types. Meta category value collections can be hierarchically organized. The specifically assigned meta category value becomes an explicit part of a data element's name.

**Mission Organization Function Person Role.** A Mission Organization Function Person Role is the assignment of a person to a particular function within an organization as it accomplishes a mission. Once assigned, the role can be described.

**Mission Organization Functional Ranked Information Need**. Once the functional areas of an organization that is accomplishing a mission of the enterprise have been created within the metabase module, Mission Organization Function Person Assignment, and after the Information Needs have been created, a set of information needs are assigned to the Mission Organization Function. Once assigned, their rank is defaulted to unknown. This needs to be changed by reviewing each mission organization function ranked information need and adjusting its rank.

**Mission Resource**. A Mission Resources is the association of a resource with a mission. Each mission may be related with one or more resources and each resource may be related to one or more missions.

**Mission-Organization.** A Mission-Organization is the association of an organization with a mission. There can be multiple organizations associated with a mission and an organization can be associated with multiple missions. The description contained within the Mission-Organization may be more refined than the description contained in either the mission or the organization.

**Mission-Organization-Function.** A Mission-Organization-Function is the association of a mission-organization with a function. A mission-organization can be associated with multiple functions and a function can be associated with multiple mission-organizations. One or more mission-organization-functions may be associated with a business information system. When they are, business events are created. This process is accomplished within the Business Information Systems module.

**Organization.** An Organization is a unit within an enterprise. It is hierarchically so any quantity of organizational levels can be represented.

**Person**. A Person is someone of interest to this metabase module.

**Predominant User.** The Predominant User meta entity is intended to distinguish among the types of users of the business information system. The example contained in the Reference data segregates users by the gross business level they represent.

**Ranking**. Once a collection of Mission Organization Ranked Information Needs have been made, it then represents the set of information needs determined important for that functional area of an organization that is accomplishing a mission of the enterprise. At the outset, all the ranks are defaulted to Unknown. That is, the ranking is unknown. That needs to be changed by first creating a set of rankings such as bottom 50%, mid 40%, and top 10%. This set of rankings are then employed to characterize the Mission Organization Ranked Information Needs

**Resource Life Cycle Node Business Information Systems Assignment**. A Resource Life Cycle Node Business Information Systems Assignment is an association between a resource life cycle node and a business information system. A business information system may be assigned to one or more resource life cycle nodes and a resource life cycle node may be related to one or more business information systems.

**Resource Life Cycle Node Business Information System Assignment**. The Resource Life Cycle Node Business Information System Assignment meta entity represents the association of the business information system with one or more resource life cycle nodes.

**Resource Life Cycle Node Database Assignment.** A Resource Life Cycle Node Database Assignment is an association between a resource life cycle node and a database. A database may be assigned to one or more resource life cycle nodes and a resource life cycle node may be related to one or more databases.

**Resource Life Cycle Node Information Need Assignment**. A Resource Life Cycle Node Information Need Assignment is the association of one resource life cycle node and an information need. A resource life cycle node may be related to one or more information needs and an information need may be related to one or more resource life cycle nodes.

**Resource Life Cycle Node Structure.** The Resource Life Cycle Node Structure is the association of one resource life cycle node an another. The association represents a relationship between the two resources for some purpose.

**Resource Life Cycle Node**. A Resource Life Cycle Node is a life cycle state within the resource. If the resource is employee then the life cycle node may be employee requisition, employee candidate, employee new hire, assigned employee, reviewed employee, and separated employee.

**Resource Life Cycle Node Structure Type.** A Resource Life Cycle Node Structure Type is a classification of a set of resource life cycle node structures that explain the collection. An example might be enablement and the associated resource life cycle node structures might related a recognized receivable resource life cycle node from the receivables resource “enables” a paid invoice resource life cycle node from the invoice resource.

**Resource Type**. A Resource Type represents a collection of resources. For example, finance resource resources, or property resources.

**Resource.** A Resource is an enduring asset of value to the enterprise. Included for example are facilities, assets, staffs, money, even abstract concepts like reputation. If a resource is missing then the enterprise is incomplete.

**Schema.** Schemas represent a database structure of tables and relationships within the enterprise. Implemented data models data models are cast within the domain of a schema. The set of all tables within a schema is not required to be taken from a single set of entities within a subject area.

**SQL Data Type**. SQL Data Type is a classification of the values represented by a column of a row of data. The data types common represented are character, integer, binary, and the like. Each SQL data type imposes a set of rules regarding allowable values and allowed operations on the values. For example adding an integer value to a date value, but disallowing the adding of two dates.

**Subject.** Subjects represent a cohesive area of discipline within the enterprise. Subjects can be hierarchical and likely match the essential resources of the business. Specified data models are cast within the domain of a subject area.

**Table Candidate Key & Column**. Table candidate key & columns are the relationship between an table candidate key and the columns that comprise the key. The columns exist within a specified sequence. Candidate key columns are not allowed to include any columns within the table’s primary key.

**Table Candidate Key.** Table Candidate Keys represent a collection of columns within an table that when their values are collectively employed would result in the retrieval or update of a single row of data for that table. There may be multiple candidate keys within an table. Columns of candidate keys are not allowed to overlap each other or the table’s primary key.

**Table Foreign Key & Column.** Table Foreign Key & Columns are the relationship between an table foreign key and the columns that comprise the key. The columns exist within a specified sequence. Foreign key columns are not allowed to include any columns within the table’s primary key.

**Table Foreign Key**. Table foreign keys represent a related table's primary key. The name of the foreign key should match closely the relationship that the key is to represent. The columns of the foreign key should be able to be deleted entirely from the table without any loss of policy. The columns of the foreign key are not allowed to overlap the columns of the table's primary key. In addition to the foreign key's columns there are additional rules governing inserts, updates, and deletes.

**Table Primary Key & Column**. Table Primary Key & Column are the relationship between an table candidate key and the columns that comprise the key. The columns exist within a specified sequence. Candidate key columns are not allowed to include any columns within the table's primary key.

**Table Primary Key.** Table Primary Keys represent a collection of columns within an table that when their values are collectively employed would result in the retrieval or update of a single row of data for that table if that table had actually been a table. There can only be one primary key within an table. Columns of primary key are not allowed to overlap each other or the table's candidate key.

**Table.** A Table is intended to be a well defined expression of one policy within a schema. Ideally, the collection of all the tables within a schema area should define a coherent collection policy. Although unlikely, some tables and even some schemas may never be represented within operational data models. Additionally, some columns within an table may never be employed. A table may contains columns that map to attributes from multiple entities. Tables can be sub-typed.

**Value Domain Data Type**. Value Domain Data Types is a classification of the values represented by a set of data values. The data types common represented are character, integer, binary, and the like. Each data type imposes a set of rules regarding allowable values and allowed operations on the values. For example adding an integer value to a date value, but disallowing the adding of two date.

**Value Domain Structure Type.** Value Domain Structure Type represent the classification of structure among a set of data element concepts.

**Value Domain Structure.** Value Domain Structure represent a collection of data element concepts that are related and are in turn represented by a collective data element concept.

**Value Domain Value Structure Type**. Value Domain Value Structure Types represent the hierarchical classification of the mapping of value domain values. For example, Gender Value Mappings then Male Gender Value Mappings and Female Gender Value Mappings then the actual male gender values that are mapped and the female gender values that are mapped.

**Value Domain Value Structure.** Value Domain Value Structures represent the mapping between values of different value domains.

**Value Domain Value.** Value Domain Values represent the actual values that are defined within the context of a value domain.

**Value Domain.** Value Domains are the included, excluded, discrete or range sets of values that are allowed within the overall context of the value meanings specified within conceptual value domains. Value domains are then either used by data elements or subsets are employed by attributes of entities, columns of tables, or DBMS columns of DBMS tables.

**View Column & Compound Data Element.** View Column & Compound Data Elements are the relationship between a view column and one or more compound data elements. The view columns & compound data elements exist within a specified sequence.

**View Column & DBMS Column.** View Column & DBMS Columns are the relationship between a view column and one or more DBMS columns. The view columns & DBMS columns exist within a specified sequence.

**View Column & Derived Data Element.** View Column & Derived Data Elements are the relationship between a view column and one or more derived data elements. The view columns & derived data elements exist within a specified sequence.

**View Column Structure Process**. View Column Structure Process is the pseudo code that defines the relationship among all the view column referenced in a set of view column structure records.

**View Column Structure Type**. View Column Structure Type represents the classification of structure among a set of view column structures.

**View Column Structure.** View Column Structure is the relationship between two view columns of different views that are governed by a specific view column structure process.

**View Column**. A View Column is mapped to one or more compound data elements, or one or more derived data elements, and one or more DBMS column. A view column also forms the basis for the interrelationship among views.

**View**. Views consist of view columns.

## 5.2 Application Architecture

The metadata management system flows from the enumerated set of metadata object classes within the previous section. Each metadata repository subsystem operates on a subset of the metadata. A brief description of each inferred metadata subsystem follows:

An **Application View Data Model Metadata System Component** enables the generation, inventory, and maintenance of the views that an application system has of the databases to which it is loading data, retrieving data and/maintaining data. This metadata then enables a good knowledge of the uses of various databases within the various business cycles and calendars through the execution of the business information systems that contain the views. Since application views are interrelated with business events and calendars it is then possible to view database processing within the context of business information systems, business calendars, and/or business cycles.

A **Business Calendars Metadata System Component** enables the creation and interrelationship of the various business calendars that govern the accomplishment of business information systems within various business cycles.

A **Business Events Metadata System Component** enables the identification and interrelationship of the various business events that occur within the accomplishment of functions and then for each business event the various collections of business information systems that are executed in support of that particular business event.

A **Business Information Systems Metadata System Component** enables the identification and interrelationship of various business information systems and their components to the application views that reference the databases upon which the business information systems act, and the business events that act as the triggers for the systems. Through these relationships the various business events along with their business cycles and calendars can be listed to then know of processing loads for each business information system.

**A Specified Data Models Metadata System Component** enables the exposition of the various specified data models that contribute specified subjects, entities and/or attributes to the development of one or more implemented data models that are to be implemented within the enterprise. Because specified data models exist at a level of abstraction higher than implemented data models they function as a coalescing mechanism for the use of the different data concepts employed within the implemented data models. These models server as collection of data model templates available for use in the construction of implemented databases. Because the specified data model is a level of abstraction lower than a 11179 Data Element metadata model each specified entity's attributes represents a deployment of the complete set of semantics of that 11179 data element. The specified data model metadata system component is supported by a full set of data modeling creation and re-engineering facilities including the importing and exporting of SQL DDL. It enables enterprise uses to view specified database models individually or across the enterprise.

A **Database Domains Metadata System Component** enables the full exposition of the data classes that exist within the context of a mission.

A **Database Object Classes Metadata System Component** enables the full specification of the data and processes that are contained within the DBMS layer of any modern database application environment. Included within the database object classes are its data structure that comprise the data segments of the database object class, the processes that create, modify, or delete rows of data of the database object tables, the states through which the database objects are transformed, and the database object information systems that transform the database objects from one valid state to the next.

A **Databases Metadata System Component** enables the visibility of the various databases of the various database architecture classes (i.e., original data capture, transaction data staging area, operational data store, data warehouses, and reference data) and their attendant schema based data model views, along with the associated business information systems that are supporting the functions of the enterprise through the various business calendars and events.

A **Functions Metadata System Component** enables the enumeration of the various functional hierarchies and commonly accepted variants of the business functions that represent the accomplishment of knowledge work by various organizations in their performance of the enterprise's mission.

An **Information Need and Characterizations Metadata System Component** enables the identification and characterization of the various information needs of the enterprise. Information needs are then interrelated with the various functions, organizations, and missions so that they can be viewed together.

An **ISO 11179 Data Element Metadata System Component** enables the creation of the various business fact templates and their semantics that are then employed to regularize all the attributes of entities and columns of tables. Include are the various components of the 11179 standard including concepts, data element concepts, data elements, conceptual value domains, value domains, and value domain values. Collectively, when interrelated with all the other data-based metadata enables data standardization and sharing across all the various database architecture classes and database applications that operate on these databases.

A **Implemented Data Models Metadata System Component** defines the various databases that are to be implemented within the enterprise. Each such implemented data model has yet to be transformed into the design required by a particular SQL DBMS. Each implemented data model consists of tables, columns, and relationships. Each column is related to its 11179 data element and to its appropriate specified data model entity attribute. Implemented data models can be boot-strapped into existence through specified data model entity, entity-set, or attribute imports. Conversely, specified data models can be built through the promotion of a implemented data model. Implemented data model table column value domains may be restricted by valid value lists, ranges, and/or excluded value lists. Within an large functional area of an enterprise there may be several dozen original data capture databases, a large quantity of TDSA databases depending on their architecture, a dozen or so operational data store databases, a similar quantity of data warehouse databases, any number of data marts, a few reference data databases depending on decisions regarding distribution. The implemented data model metadata system component is supported by a full set of data model creation and re-engineering facilities including the importing and exporting of SQL DDL. It enables enterprise uses to view implemented database models individually or across the enterprise.

A **Missions Metadata System Component** enables the identification and definition of the set of missions that are undertaken by the enterprise. Once identified these would be able to be interrelated with the appropriate database domains, functions, and organization, and through other relationships to know of the various databases and business information systems that operate on various business events and cycles.

An **Organizations Metadata System Component** enables the incorporation of the various organizations that exist within the enterprise and the interrelationship with enterprise missions functions, business events and calendars and their associated business information systems. These enable the full exposition of the activities of various organizations via their functions and business information systems.

A **Persons and Roles Metadata System Component** enables the capture of the various staff that exist within the enterprise and the roles they play within functions and organizations.

A **Operational Data Models Metadata System Component** enables the creation of the actual DBMS-based data models that are then compiled and are operating with the business information systems to collect, store, evolve and report enterprise data. These operational data models consist of its database reference, DBMS schema, DBMS tables, DBMS columns, and relationships. The operational data models can be boot-strapped into existence through implemented data model table, table-set, or column imports. The operational data models are interrelated directly with the application view models and also with their implemented data models. The operational data model metadata system component is supported by a full set of data model creation and re-engineering facilities including the importing and exporting of SQL DDL. It enables enterprise uses to view operational database models individually or across the enterprise or within the context of implemented data models.

A **Resources and Life Cycles Metadata System Component** enables the identification of the various resources within the enterprise that collectively represent either the infrastructure or external product set of the enterprise. Infrastructure resources include for example, staff, facilities, contracts, finance, and the like. External products include manufactured products, services to customers, and the like. Each resource is then defined in terms of its life cycle. Resource life cycle nodes from different life cycles are interrelated to show enterprise-based interdependencies. Databases and Business information systems, and information needs are then interrelated to each life cycle node. Collectively the fully attributed resource life cycles enable the enterprise to view its complete operation in terms of its essential resources that define its very existence.

## 5.3 Technical Architecture

The technical architecture of any database application consists of an enumeration of the characteristics of its logical database, physical database, interrogation, system control, and computing infrastructure operating environment.

The characteristics of the logical database of a comprehensive metadata management system include:

- Each meta-entity is expressed as a separate ANSI standard SQL table.
- All relationships are expressed as traditional ANSI standard SQL relationships.
- All referential integrity and referential actions are schema based.
- Column and table constraints as well as assertions and triggers are SQL based.
- All SQL statements are not vendor proprietary.

The characteristics of the physical database include:

- All metadata is loadable through either vendor provided 4GL, or 3GLs, or through an ODBC access metadata application presentation layer.

- Supported by a process that can read SQL DDL for use in creating data models at the conceptual, logical, and physical levels.

- Able to export XML wrapped metadata through DBMS vendor and/or metadata application provided facilities.

The characteristics of interrogation include:

- Ability to report directly from the metadata database's explicit SQL schema through report writers provided by the SQL Vendor or through third party vendors such as Crystal Reports.

- Ability to publish metadata in HTML for the Internet.

The characteristics of system control include:

- Supported by the SQL DBMSs facilities for at least audit trails, transaction rollback, logical and physical database re-organization, and security and privacy.

The characteristics of the operating environment include:

- Operational on any MS/Windows operating system

- Able to place the meta data database on a server running under either a Windows O/S or Unix

- Operational under any commonly available SQL based DBMS through ODBC.

## 6.0 Use Scenarios

A comprehensive metadata management system can either be a passive repository for accomplished knowledge work or can be integral component of accomplishing knowledge work. Clearly the later is preferred as the population and use of the facility cannot then be ignored. If the policy is made that a deliverable exists only after it is able to be retrieved from the metadata repository, and that corrections or revisions of deliverables are accomplished only when they are retrieved from the metadata repository, then the repository will certainly take on a critical, central, and active role within any knowledge work project environment. With that as a given, the following are typical use scenarios for a comprehensive metadata repository and its attendant metadata system:

- Build, maintain, and employ business cycles, calenders and interrelate business information system execution cycles
- Build, maintain, and employ business information system specifications
- Build, maintain, and employ conceptual data models
- Build, maintain, and employ database application projects
- Build, maintain, and employ database domain models
- Build, maintain, and employ database object classes
- Build, maintain, and employ function models
- Build, maintain, and employ information needs analysis
- Build, maintain, and employ information systems plans
- Build, maintain, and employ ISO 11179 data elements and supporting metadata
- Build, maintain, and employ logical data models
- Build, maintain, and employ mission models.
- Build, maintain, and employ organization models
- Build, maintain, and employ physical data models
- Build, maintain, and employ resource life cycles

Each scenario is briefly described.

**Build, Maintain, and Employ Business Cycles, Calenders and Interrelate Business Information System Execution Cycles.** The actual workflow of a collection of business information systems exists within business cycle, business events, and business calendars. Each business cycle is defined so that the sequence for the accomplishment of business information systems is clear. Each business information system is then activated by the business event that is associated with the business cycle. Business calendars need to also be defined as they may contain specific days on which certain processes must be completed or cannot occur. Business calendars must be interrelated with business cycles.

**Build, Maintain, and Employ Business Information System Specifications.** Each business information system specification is hierarchical and thus includes subsystems. Each subsystem is named and generally described as to it content and purpose. The levels of detail, for example pseudo code for business information system modules are purposely omitted because that is best left to database information system development environments. If that level of detail were in the metadata repository then there would be a 100% likelihood that it would out of synch with the actual business information system. Business information systems are integrated with the database object classes that they invoke to transform database objects from one state to the next and are also integrated with the business events that in the name of the business function cause the execution of the business information system.

**Build, Maintain, and Employ Specified Data Models.** Specified data models are collections of entities, attributes and relationships that can be used as data model templates for implemented databases. Each entity within a specified model should be the data specification of a well defined policy within the enterprise. A collection of entities within a particular subject should conform to a larger and more complex policy. A implemented database is bounded by schema and is intended to be implemented by a particular DBMS thus arising in an operational database that collects, stores, and maintains actual business data. In contrast, the specified database's entities are bounded only by the subject within which it is defined. In the construction of a implemented data model, one more entities may contribute attributes to form the column of the implemented data model's tables. Specified data models enable the creation of standard data structures that when employed in a implemented data model ensure completeness, rigor, and the data standardization essential for data sharing. The semantics of attributes of a specified data model entity are derived from ISO 11179 data elements. In total, the ISO 11179 data elements, specified data models, implemented data models, and operational data models all form a general hierarchy of business facts within the enterprise that enable a clear picture of where and how all business facts are defined and deployed. Specified data models can be created inductively through the promotion of a single implemented data model to the specified data model level. Then, data modeling activities would occur to break apart the specified data model into individual subjects and collections of entities within those subjects. Entities can be interrelated across subject areas to represent specified data model factoring.

**Build, Maintain, and Employ Database Application Projects.** Each database application project consists of a work plan, deliverables, assigned staff and a work environment. As projects are proposed they are set within the context of information systems plans. Each project's metadata is linked to the actual deliverable's metadata so they can be reviewed to better understand the work accomplished. As work is performed, work-accomplishment time-cards are entered so that earned value reporting is automatically produced. Since the projects would have been estimated via standard metrics, the actual accomplishments can be used to adjust the metrics. Finally, since all projects exist within the metadata repository they can be viewed and analyzed collectively or individually, or in groups of contained project tasks.

**Build, Maintain, and Employ Database Domain Models.** Database domains are "noun-intensive" descriptions of the data that is inferred by the lowest level of a mission hierarchy. Each database domain is thus restricted in scope to that of the mission leaf. Additionally, each database domain is represented by a simple entity-relationship diagram (ERD). When all the relevant database domains are completed their ERDs are combined to ensure that the entities that are named the same are in fact the same and are represented at the same level of granularity.

**Build, Maintain, and Employ Database Object Classes.** Database object classes are the encapsulated data structures, processes, and constraints necessary to transform a set of data from one value state to the next. Database object classes are essential to the integrity of databases. In modern SQL DBMSs, database object classes are largely able to be constructed through the use of persistent views that map to a collection of columns across a set of tables. The value state integrity is governed by columns and table constraints. The value states are transformed through stored procedures within assertions and triggers. It is important to define database object classes within the domain of the DBMS to ensure that all external language agents such as 4GLs, query languages, and 3GLs are forced to proceed through these DBMS defined and encapsulated database object classes.

**Build, Maintain, and Employ Function Models**. As a database project commences, it is important to know just what role it will play within the manual functions that are accomplished by any organization within the scope of a mission. The hierarchical function models are created and interrelated with the various organizations that perform them. Because functions are human activities, there may be multiples sets of functions that are generally equivalent but differ in style of knowledge worker processes. The differences are not critical because the relationship between a business information system and a business function is through the intermediary, business event.

**Build, Maintain, and Employ Information Needs Analysis.** As a database project is started, it is important to know just what are the information needs that are to be encompassed within the database design. The information needs are thus gathered and stored into the metadata repository along with their characteristics such as timeliness, granularity, production needs, and the like. The information needs are interrelated with both the functions that are being supported by the information needs, and the resource life cycle nodes for which the information needs essentially become the work product evidences of the resource life cycle node state.

**Build, Maintain, and Employ Information Systems Plans.** Every project within an enterprise commonly requires the specification and implementation of multiple information systems. Within an enterprise as a whole there may be hundreds of information systems being planned. A comprehensive information system plan sets all the information systems within the context of the resource life cycle nodes, and then estimates their duration via standardized project methodologies and standard metrics. This enables the enterprise to view all its projects, and to know the effects of accelerating and/or delaying any particular project.

**Build, Maintain, and Employ Iso 11179 Data Elements and Supporting Metadata.** Attributes of entities and columns of tables should all draw their semantics from data elements. A data element is a context independent (i.e., entity and/or table independent) business fact semantic template. It is well accepted practice that the quantity of data elements are a small fraction of attributes and/or columns. Supporting data elements are multiple higher levels of data element metadata including concepts, conceptual value domains, value domains and sets of values. The values sets can be directly allocated to DBMS schema columns as constraints. More likely they would form the rows of data within the reference data database.

**Build, Maintain, and Employ Implemented Data Models**. A implemented data model is a collection of tables, columns, and relationships bounded by a schema. Implemented data models are built as a precursor to the design of the database object classes that operate to maintain data integrity and value transformations. It is common to build a implemented database within the scope of a reasonably large mission hierarchy such as human resources, finance, facilities, customers, sales management, distribution, or inventory. Database object classes are transformed through business information systems. Implemented databases commonly conform to particular data architecture classes such as original data collection, transaction data staging area (TDSA), data warehouses, data marts, and reference data databases. Implemented database table columns should all be derived from attributes from entities of one or more specified data models. Implemented data models also act as the "parent" of one or more operational data models. In total, the ISO 11179 data elements, specified data models, implemented data models, and operational data models all form a general hierarchy of business facts within the enterprise that enable a clear picture of where and how all business facts are defined and deployed. Implemented data models can be created inductively through operational data model imports that exist within a certain scope, and then through the promotion of a single operational data model to the implemented data model level. Then, data modeling activities would occur to expand the scope of the implemented data model to be that of the union of all the operational data models.

**Build, Maintain, and Employ Mission Models.** The mission models are the boundaries of the scope of the enterprise. It is within mission models that database domains that lead to database designs are created. Missions are also the scope boundaries for all enterprise organizations and functions.

**Build, Maintain, and Employ Organization Models.** As a database project commences An organization model is built to then allocated to the various missions and functions. This permits the easy identification of those components of the enterprise that are involved in any database project effort.

**Build, Maintain, and Employ Operational Data Models.** The operational database is a implemented database that may have been subsetted and/or transformed to server the particular needs of a DBMS, or performance requirement. Operational databases are mapped back to their "parent" implemented models through a column (implemented data model) to DBMS column (operational data model) mapping. Operational data models, are "hosts" to the various SQL views that in turn act as intermediaries to the business information systems that access the databases. There may be multiple transformations of a particular implemented database, and each exists and is mapped back to its "parent" implemented database. In total, the ISO 11179 data elements, specified data models, implemented data models, and operational data models all form a general hierarchy of business facts within the enterprise that enable a clear picture of where and how all business facts are defined and deployed.

**Build, Maintain, and Employ Resource Life Cycles.** Enterprises can be viewed as a collection of resources that are moved through well defined life cycles. In this context, resources, some concrete and some abstract would include staff, finance such as payables, receivables, and payroll, as well as facilities, contracts, customers, sales management, distribution, products, manufacturing lines, inventory, missions, functions, organizations, reputation and the like. Each life cycle node of a resource, for example the recognition of a receivable is commonly supported either by manual or automated systems and databases. A resource life cycle node from one resource life cycle can be related to a node on another life cycle as a way of facilitating the related-to node. For example, Issuance of a contract from within the contacts resource facilitates the recognition of the receivable within the receivables resource. This interdependence enables the enterprise, as a whole, to be seen as an network of interconnected resources that has to function effectively as a complete system for the enterprise to be successful. Assisting in the effectiveness of a given resource life cycle node of a resource are the databases and information systems that assist persons who are performing functions within their organizations in support of the enterprise's mission.