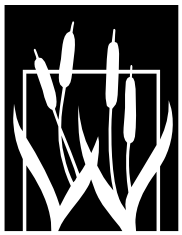

Whitemarsh
Information Systems Corporation

Engineering and Managing Information Systems Plans

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

Table of Contents

Figures

Tables

1.0 Objective

The objective of this paper is to present the Whitemarsh approach to the practical and efficient creation of Information Systems Plans that can, as a collection bet set within enterprise architectures, and can be managed, re-engineered, re-prioritized, and recast as technology and business needs evolve.

2.0 Topics Covered

The topics is this paper include:

- Rationale for an Information Systems Plan
- Successful Information Systems Plans
- Summary

3.0 Rationale for an Information Systems Plan

Every year corporations commonly spend about 5% of their gross income on information systems and their supports. For a $300 to $700 million dollar corporation that's from about $15,000,000 to $35,000,000! A significant part of those funds support enterprise database, a philosophy of database system applications that enable corporations to research the past, control the present, and plan for the future.

Even though an information system costs from $1,000,000 to $10,000,000, and even through most chief information officers (CIOs) can specify exactly how much money is being spent for hardware, software, and staff, CIOs cannot however state with any degree of certainty why one system is being done this year versus next, why it is being done ahead of another, or finally, why it is being done at all.

Enterprises do not have model-based information systems development environments that allow system designers to see the benefits of rearranging an information systems development schedule. Consequently, the following questions cannot be answered:

- What effect will there be on the overall schedule if an information system is purchased versus developed?

- At what point does it pay to hire an abnormal quantity of contract staff to advance a schedule?

- What is the long term benefit from 4GL versus 3GL?

- Is it better to generate 3GL than to generate/use a 4GL?

- What are the real costs of distributed software development over centralized development?

If these questions were transformed and applied to any other component of a business (e.g., accounting, manufacturing, distribution and marketing), and remained unanswered, that unit's manager would surely be fired!

We not only need answers to these questions NOW!, we also need them quickly, cost effectively, and in a form that they can be modeled and changed in response to unfolding realities. This paper provides strategies for developing answers to these questions. Under this same title, Whitemarsh provide a book, course, and the Metabase System fully supports Information Systems Plan software creation.

Too many half-billion dollar organizations have only a vague notion of the names and interactions of the existing and under development information systems. Whenever they need to know, a meeting is held among the critical *few*, an inventory is taken, interactions confirmed, and accomplishment schedules are updated.

This ad hoc Information Systems Plan was possible only because all design and development was centralized, the only computer was a main-frame, and the past was acceptable prologue because budgets were ever increasing, schedules always slipping, and information was not yet part of the corporation's critical edge.

Well, today is different, really different! Budgets are decreasing, and slipped schedules are being cited as preventing business alternatives. Confounding the computing environment are different operating systems, DBMSs, development tools, telecommunications (Lan, Wan, Intra-, Inter-, and Extra-net), and distributed hard- and software.

Rather than having centralized, long-range planning and management activities that address these problems, today's business units are using readily available tools to design and build ad hoc stop-gap solutions. These ad hoc systems not only do not interconnect, support common semantics, or provide synchronized views of critical corporate policy, they are soon to form the almost impossible to comprehend confusion of systems and data from which systems order and semantic harmony must spring.

Not only has the computing landscape become profoundly different and more difficult to comprehend, the need for just the right--and correct--information at just the right time is escalating. Late or wrong information is worse than no information.

Information systems managers need a model of their information systems environment. A model that is malleable. As new requirements are discovered, budgets modified, new hardware/software introduced, this model must be such that it can reconstitute the Information Systems Plan in a timely and efficient manner.

3.1 Characteristics of a Quality Information Systems Plan

A quality Information Systems Plan must exhibit five distinct characteristics before it is useful. These five are presented Table 1.

Characteristic	Description
Timely	The Information Systems Plan must be timely. An Information Systems Plan that is created long after it is needed is useless. In almost all cases, it makes no sense to take longer to plan work than to perform the work planned.
Useable	The Information Systems Plan must be useable. It must be so for all the projects as well as for each project. The Information Systems Plan should exist in sections that once adopted can be parceled out to project managers and immediately started.
Maintainable	The Information Systems Plan should be maintainable. New business opportunities, new computers, business mergers, etc. all affect the Information Systems Plan. The Information Systems Plan must support quick changes to the estimates, technologies employed, and possibly even to the fundamental project sequences. Once these changes are accomplished, the new Information Systems Plan should be just a few computer program executions away.
Quality	While the Information Systems Plan must be a quality product, no Information Systems Plan is ever perfect on the first try. As the Information Systems Plan is executed, the metrics employed to derive the individual project estimates become refined as a consequence of new hardware technologies, code generators, techniques, or faster working staff. As these changes occur, their effects should be installable into the data that supports Information Systems Plan computation. In short, the Information Systems Plan is a living document. It should be updated with every technology event, and certainly no less often than quarterly.
Reproducible	The Information Systems Plan must be reproducible. That is, when its development activities are performed by any other staff, the Information Systems Plan produced should essentially be the same. The Information Systems Plan should not significantly vary by staff assigned.

Table 1. Quality Information Systems Plan Characteristics.

Whenever a proposal for the development of an Information Systems Plan is created it must be assessed against these five characteristics. If any fail or not addressed in an optimum way, the entire set of funds for the development of an Information Systems Plan is risked.

3.2 Comparison of Three Information Systems Plan Development Approaches

The three traditional approaches to the development of an Information Systems Plan are:

- IBM's Business System Plan
- James Martin's Strategic Data Planning
- Clive Finklestein's Strategic Management Plan

Table 2 presents a summary of the approaches. A full explanation of the three approaches including summary work plans is provided in the Information Systems Plan book from the Whitemarsh website.

Characteristic	Approach Name		
	IBM's Business System Plan	**James Martin's Strategic Data Planning**	**Finklestein's Strategic Management Plan**
Timely	20 staff years	16 staff years	22 staff years
Useable	Low	Low	High
Maintainable	Low	Medium	Medium
Quality	Medium	Medium	High
Reproducible	Low	Low	Medium

Table 2. Traditional Information Systems Plan Characteristics.

It should be easy to conclude that these three approaches are not acceptable for an enterprise that must develop five-year strategic, three-year tactical, and then one-year operational updates to its Information Systems Plan.

Few enterprises either complete, or once completed, maintain Information Systems Plans. An analysis of the details behind the three traditional approaches through which Information Systems Plans are attempted clearly reveals why they take so long, do not meet the required level of quality, and finally are not cost-effective. Key among the reasons are:

- IBM's and Martin's (BSP and SDP) methodology requires a full function decomposition as well as a high level data model. In addition, the functional decompositions must be finished before the Information Systems Plan can be started.

- Finklestein's methodology (SMP) requires the development of a fifth-normal form data model and an almost complete process model prior to the start of the Information Systems Plan.

It is very difficult to obtain consensus on a functional decomposition for any one application, much less across all the information systems within the entire corporation. That is because functional analysis requires identification and codification of how activities are performed. In short, the codification of style. This type of analysis leads to conflicts, power struggles, and endless nit-picking. In the end, nobody likes the results.

Once, or if ever completed, both IBM and Martin use the resulting Information Systems Plan as a foundation for identifying information systems. Building the Information Systems Plan on top of such a foundation of discord cannot possibly result in stable, enduring information systems.

Finklestein's methodology requires the development of a fifth-normal form data model for the entire enterprise. Such an extraordinarily detailed effort certainly embraces three to five thousand entities, all the appropriate attributes, full definitions for entities and attributes, and a full exposition of every relationship among all entities. While development of such a edifice is a valuable final shrine once all the identified information systems have been implemented, it is totally unnecessary for the Information Systems Plan.

Even if the length of time for IBM, Martin, and Finkelstein was deemed acceptable, produced plans would still unacceptable because:

- The form of their Information Systems Plans is a static set of matrices that merely show associations and hierarchies of data and process.

- Each identified information system is not tied to existing information systems and data for contextual assessment and analysis

- The identified information systems are not estimated through any standard work break downs structures (standardize project methodology templates) and standard metrics enabling development of alternative time-lines, costs and resource loadings.

- The identified set of information systems is not placed in any precedence sequence that supports a rational, dependable project sequence that can be loaded into a project management package for computation of Gantt and CPM outputs.

Because of all these inadequacies, any Information Systems Plan developed through the IBM, Martin, and Finkelstein techniques produces a motionless, unchangeable blueprint. Great for illustrating the past, for controlling an instant in the present, but poor for modeling the future.

3.3 Whitemarsh's Information Systems Plan: A Difference in Kind

Whitemarsh's Information Systems Plan approach was built directly on the strengths and designed-out the weaknesses of the three traditional approaches[1]. The Whitemarsh approach employs:

- A mission based foundation for the Information Systems Plan rather than the extremely politically charged function modeling.
- A data driven approach for the formulation of all databases, but only at a high level. That is, database objects.
- The Whitemarsh Metabase System to store, evolve, report and analyze all collected analysis products.
- Project, deliverable, and metric templates to make project estimation accurate and reproducible.
- Ron Ross's interrelated resource life-cycles as a lattice upon which all Information Systems Plan proposed projects are cast.

As a consequence of this difference in kind, the Whitemarsh Information Systems Plan exhibits these characteristics:

Whitemarsh Information Systems Plan Approach	
Characteristic	**Description**
Timely	Creation of the Whitemarsh Information Systems Plan is timely because it can be created in less than three staff years. This is an order of magnitude less than IBM's BSP, Martin's SDP, or Finkelstein's SMP. The time is even less if components already exist.
Useable	The Whitemarsh Information Systems Plan enables its users to make both strategic and tactical decisions regarding business information system and database project sequencing based squarely on business fundamentals.
Maintainable	The Whitemarsh Information Systems Plan is maintainable because it mainly uses metadata already essential for enterprise database. Metadata that should be readily available in the Metabase System.

[1] Much of the credit for the formulation of the approach upon which the Whitemarsh Information Systems Plan approach is due to Dagmar Bogan and Stan Hopkins, both formerly of The MITRE Corporation.

Whitemarsh Information Systems Plan Approach	
Characteristic	**Description**
High Quality	The Whitemarsh Information Systems Plan is a quality product because it is accomplished through common-sense-based techniques that have been proven over 20+ years.
Reproducible	The Whitemarsh Information Systems Plan is reproducible because at each review, the resources can be re-examined, new technology set into place, basic RLC precedence vectors re-cast, and then the whole plan regenerated.

Table 3. Whitemarsh Information Systems Plan Characteristics.

3.4 Context of Whitemarsh Information Systems Plan

The Whitemarsh Information Systems Plan exists squarely within the Knowledge Worker Framework that is provided in Table 4.

3.5 Information System s Planning within the Context of the Metabase System System Environment

The Whitemarsh Information Systems Plan is the plan through which databases and information systems of the enterprise are accomplished in a timely manner. A key facility through which the Information Systems Plan obtains its "data" is the Whitemarsh Metabase System. The domain of the Metabase System is illustrated in Figure 1, and, as seen through this diagram, persons through their role within an organization perform functions in the accomplishment of enterprise missions, they have information needs. These information needs reflect the state of certain enterprise resources such as finance, people, and products that are known to the enterprises. The states are created through business information systems and databases.

The majority of the metadata employed to develop the Whitemarsh Information Systems Plan resides in resource life cycles, the databases and information systems, and the project management meta entities from the Whitemarsh Metabase System component, project management. This more narrow context is presented in Figure 2.

Knowledge Worker Framework						
Perspective	**Mission**	**Database Object**	**Business Information System**	**Business Event**	**Business Function**	**Organization**
Scope	Business missions	Major business resources	Business information Systems	Interface events	Major business scenarios	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	Data Elements Specified data models and Identified Database objects	Information system designs	Invocation protocols, input and output data, and messages	Best practices, quality measures and accomplishment 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
Deployment	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 Interface data model	Operating information systems	Start, stop, and messages	Detailed procedure based instructions	Daily activity executions, and assessments

Table 4. Knowledge Worker Framework.

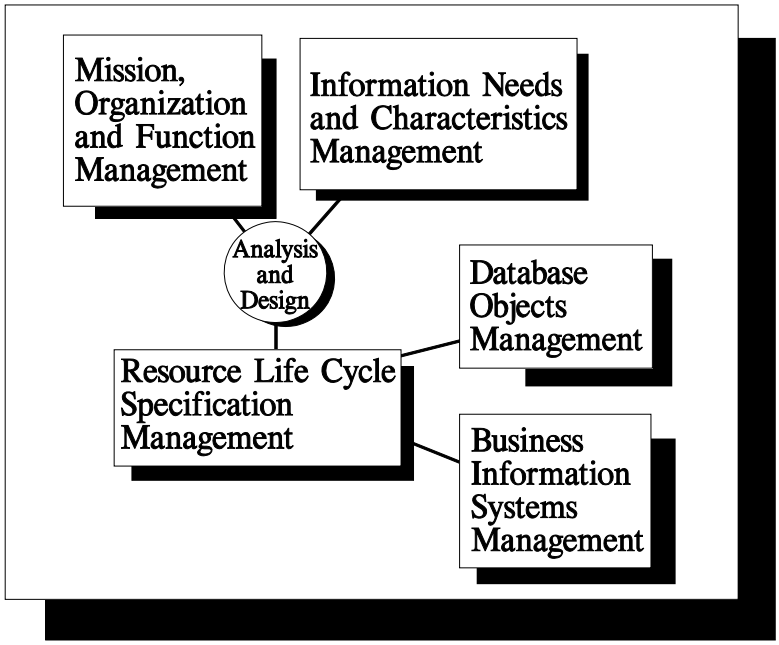

Figure 1. Metabase System domain.

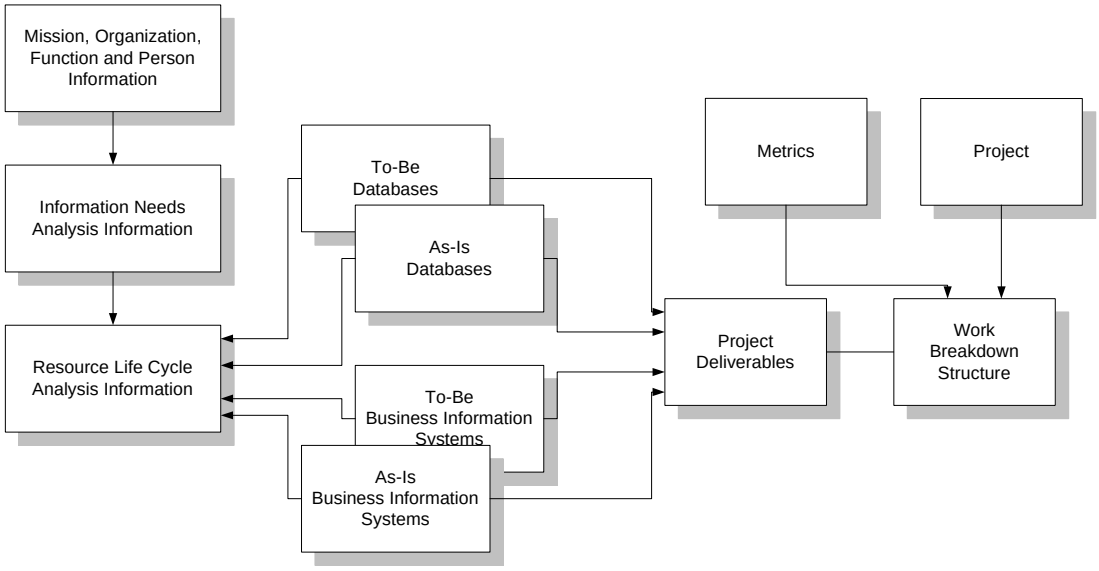

Figure 2. Meta model for Information Systems Plans.

4.0 Successful Information Systems Plans

The Information Systems Plan project described in this paper sets out the sequence for implementing specific information systems. The goal of the strategy is to deliver the most valuable business information at the earliest time possible in the most cost-effective manner.

The end product of the information systems project is an Information Systems Plan. Once deployed, the information systems department can implement the plan with confidence that they are doing the correct information systems project at the right time and in the right sequence. The focus of the Information Systems Plan is not one information system but the entire suite of information systems for the enterprise. Once developed, each identified information system is seen in context with all other information systems within the enterprise.

The following steps are involved in the development of the Whitemarsh Information Systems Plan:

1. Create the mission model
2. Develop a high-level data model
3. Create the resource life cycles (RLC) and their nodes
4. Allocate precedence vectors among RLC nodes
5. Allocate existing information systems and databases to the RLC nodes
6. Allocate standard work break down structures (WBS) to each RLC node
7. Load resources into each WBS node
8. Schedule the RLC nodes through a project management package.
9. Produce and review of the Information Systems Plan
10. Execute and adjust the Information Systems Plan through time.

Collectively, the first nine steps take about 5000 staff hours, or about $500,000. Compared to the IS budget $15-35 million, that's only about 3.0% to 1.0%.

If the pundits are to be believed, that is, that the right information at the right time is the competitive edge, then paying for an Information Systems Plan that is accurate, repeatable, and reliable is a small price indeed! The remainder of this section presents an overview of each of the ten steps.

Step 1. Create Mission Model

A critical distinction between the Whitemarsh approach and the other three approaches is that Whitemarsh's approach is founded on enterprise missions. Missions are a-political, strategic, long range, and are easily accomplished. In contrast, IBM, Finklestein and Martin are all function-based analysis, which are "politically charged," stylistic, subject to change, and because of these characteristics, are almost impossible to accomplish.

Missions and mission descriptions are represented through hierarchically composed text. They are natural and are devoid of the effects from organizational structure stylistic effects. Missions from enterprises from the same "line of business" are very similar. In contrast, their function models may be quite different because of effects imposed by management styles and organizational structures. Simply stated, mission descriptions are goal and objective oriented and are best seen as characterizations of the idealized end-results, without any regard for "who and how."

For many organizations, the mission description document often represents the first overall statement of their raison d'etre. The table that follows contains the overall mission description of the mythical Systems Engineering Corporation, and the subordinate mission description for its human resources mission.

Systems Engineering Corporation Mission Descriptions	
Mission Level	**Description**
Overall Mission	The Systems Engineering Corporation, an independent not-for-profit corporation, provides research, development, engineering, and scientific activities to a restricted set of clients in the public interest. These services are internally supported by human resources, finance, and various support services.
Human Resources subordinate mission	Human-Resources ensure that there is an appropriate quantity and quality of staff available to fulfill Systems Engineering Corporation's mission. To fulfill its mission, human resources: • Plans for and administers employee placement and recruiting plans • Develops, and administers benefits programs • Performs salary compensation and analysis • Maintains employee records • Provides services that guarantee that all employees are treated fairly and consistently • Ensures that Systems Engineering Corporation's policies and procedures comply with federal, state, and local laws • Conducts special projects.

Table 5. Mission Model for Systems Engineering Corporation.

The complete mission of the Systems Engineering Corporation spans about 35 pages. It was constructed in just two staff months. The reason it was so fast to construct was that the missions were constructed to reflect on the characteristics of the idealized end results were stripped of organization effects, and descriptions of how the missions were accomplished were removed.

It is important to distinguish between missions and functions. At first, missions and functions look very much alike. However they are not. Table 6 illustrates the key differences between Missions and Functions.

Mission versus Function Critical Differences	
Mission	**Function**
Missions are descriptions of the characteristics of the end result. Missions are noun-based sentences	Functions are descriptions of how to accomplish an end result. Functions are verb-based sentences
Missions are a-political. They are devoid of "who and how." There should only be ONE mission description for a mission.	Function hierarchies are commonly tainted by organizations and styles. There can be any number of equivalent versions of a given function.
Databases and Business Information Systems are based on missions	"Human" activities and organizations are based on business functions
When you "BPR" missions you have a different business.	When you "Business Process Re-engineering (BPR)" functions you still have the same business.
Mission descriptions are strategic and long range	Functions are tactical to operational, and medium to short range, and are organizationally sensitive

Table 6. Critical differences between Missions and Functions.

Building an Information Systems Plan on the basis of functions is a 100% guarantee of failure.

Step 2. Build the High Level Data Model

The high level data model is created in two steps: building database domains, and creating database objects.

It is critical to state that the objective of this step is the *high*-level data model. The goal is NOT to create a low level or fully attributed data model. The reasons that only a high-level data model is needed is straight forward:

- No database projects are being accomplished, hence no detailed data modeling is required

- The goal of the Information Systems Plan is to identify and resource allocate projects including database projects and for that goal, entity identification, naming and brief definitions is all that is required for estimating.

The message is simple: any money or resources expended in developing a detailed data model is wasted.

Step 2.1 Create Database Domains

Database domains are created from the "bottom" leaves of the mission description texts. There are two cases to consider. First, if the mission description's bottom leaves are very detailed, they can be considered as having being transformed into database domains. That is they will consist of lists of nouns within simple sentences. The other case is that the mission descriptions have been defined to only a few levels, and the lists of nouns that would result from the development of database domains have yet to be uncovered.

Table 7 presents the database domain for accounts payable. Whenever a database domain describes complex sets of data, multiple levels of the database domain description may be required. These subdomains are expressed as additional paragraphs. A review of these paragraphs clearly show that the text is "noun-intensive." The "who and how" is clearly missing. That is the way it should be. If the "who and how" were contained in the database domains then they would not be independent of either process or organization.

A series of diagraming techniques created especially for data and the relationships among data is called entity-relationship (ER) diagraming. Within one style of this technique, the entities are drawn as rectangles and the relationships are drawn as diamonds. The name of the relationship is inside the diamond. Another style of ER modeling is to just have named lines between the entities. Since the domain of the diagram is data, it is called the database domain diagram.

Figure 3 shows a database domain diagram for the subordinate database domain accounts payable. The database domain diagraming technique shown here is not intended to be strictly in third normal form. Rather, the diagraming technique is just supposed to depict the required entities and the relationships among the entities. It is possible that some of the entities are merely data elements, while others are complex and might have nested structures..

The purpose of the database domain diagram is not to be precise and exacting but to be comprehensive. The goal is to have the reviewer say, "that's just the right kind of data needed to satisfy the required mission description."

Database Domain Description
Accounts payable effects the satisfaction of debts of the Systems Engineering Corporation. These debts are satisfied through disbursements. Disbursements are generated as a result of payable invoices. Disbursements are paid to either an outside agency or to someone who is affiliated with the corporation. Certain payable invoices, when funds are disbursed result in fixed assets to the corporation. Payment invoices arise because of accepted receipts resulting from received shipments, accepted procurement orders against various types of contracts, scheduled payments against one or more forms of a contract. Each payable invoice consists in invoice header information and a set of one or more invoice line items. Procurement orders consist of procurement header information and one or more procurement order line items. Each procurement line item may result in one or more receipt line items from different receipts, one or more payable invoice line items. Each receipt consists of receipt header information and a set of one or more receipt lines. Payables are properly recorded against budgeted lines of an authorized budget. Disbursements result in journal entries that are recorded in the general ledger. Contracts of the corporation are the following types: service, materials/goods, or real estate. Materials or goods contracts can be for purchases or rentals from vendors. Real estate contracts can be for real estate purchases or rentals. Real estate purchase contracts may result in one or more loan agreements.

Table 7. Database Domain: Accounts Payable

When all the database domain diagrams are created, siblings are combined. Entities that are named the same are not presumed to be the same. Analysis must show that to be true. If not, one or both of the entities must have their name and definition changed. As the sets of sibling diagrams are merged from lower to higher levels, the quantity of commonly named entities on different diagrams diminishes. Diagram merger becomes optional when the use analysis of a common entity is subject to update (add, delete, or modify) in one diagram and is only referenced (read) in another diagram.

For half-billion dollar businesses, there are about 60 or so significant areas within the mission description for which database domains and collateral database domain diagrams have been constructed. These 60 make up about ten subject area databases. Across all the database domain diagrams there are about 500-600 nonredundantly defined entities. The creation of database domain diagrams as a graphic expression of the database domains is the first step in creating the enterprise data model. The second step is database object creation.

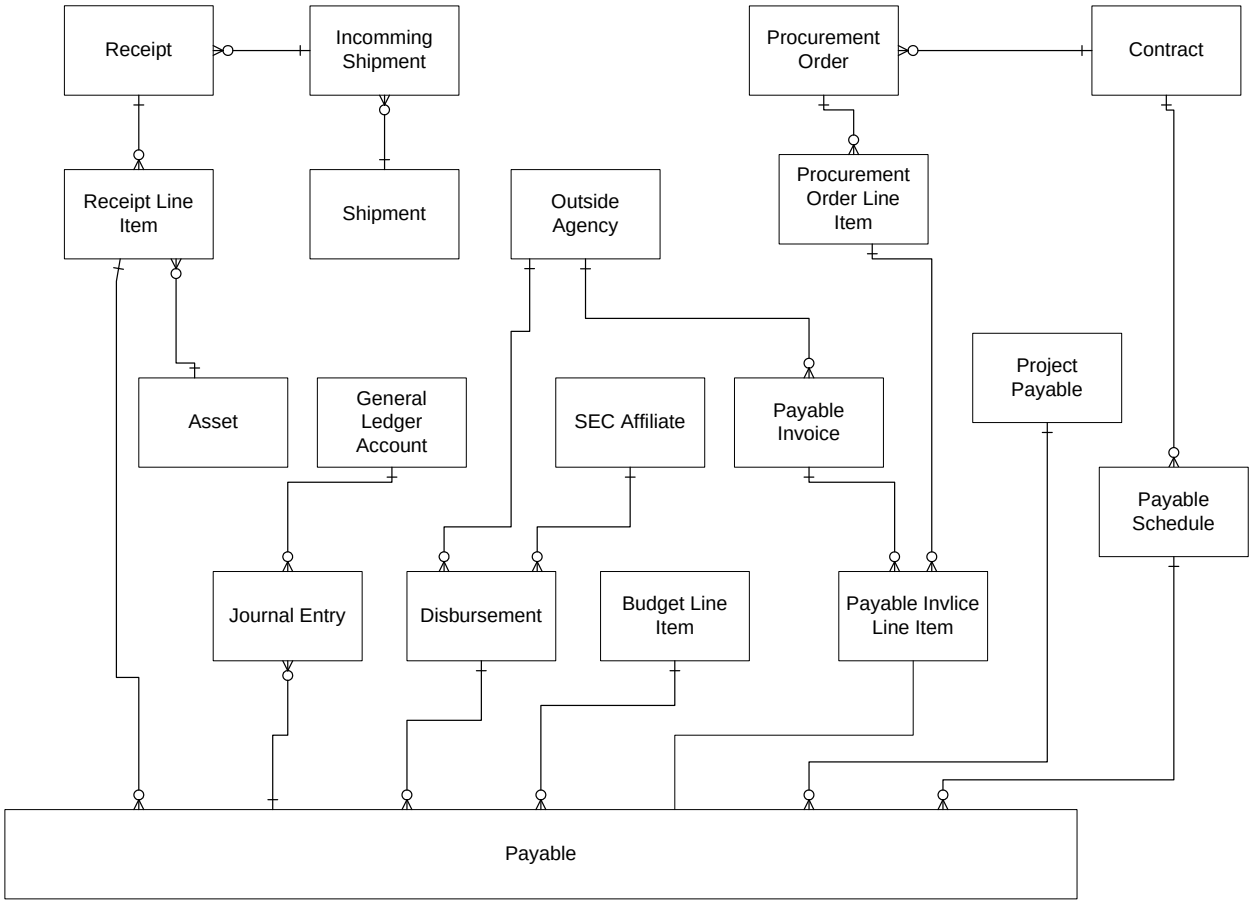

Figure 3. Accounts Payable entity relationship diagram.

Step 2.2 Define Database Objects

In today's parlance, a lucid *policy-procedure* pair is called a business object. When the policy-procedure pair are completely defined within the language constructs of ANSI/SQL and is stored, retrieved, and maintained in an ANSI/SQL database through a sequence of well-defined states, the business object is a database object. The goal of database object analysis is to enable the definition of both the data structure and the data structure transformations that:

- Creates a new database object in the database
- Transforms a database object from one coherent state to another

- Removes a database object from the database

Database objects are found by researching business policies and procedures. Database objects are however much more than just collections of policy-homogeneous entities. In fact database objects consist of four main parts:

- [database object] Data Structure: the set of data structures that map onto the different value sets for real world database objects such as an auto accident, vehicle and emergency medicine incident.

- [database object] Process: the set of database object processes that enforce the integrity of data structure fields, references between database objects and actions among contained data structure segments, the proper computer-based rules governing data structure segment insertion, modification, and deletion. For example, the proper and complete storage of an auto accident.

- [database object] Information System: the set of specifications that control, sequence, and iterate the execution of various database object processes that cause changes in database object states to achieve specific value-based states in conformance to the requirements of business policies. For example, the reception and database posting of data from business information system activities (screens, data edits, storage, interim reports, etc.) that accomplish entry of the auto accident information.

- [database object] State: The value states of a database object that represent the after-state of the successful accomplishment of one or more recognizable business events. Examples of business events are auto accident initiation, involved vehicle entry, involved person entry, and auto accident DUI (driving under the influence of alcohol/drugs) involvement. Database object state changes are initiated through named business events that are contained in business functions. The business function, auto accident investigation includes the business event, auto-accident-incident initiation, which in turn causes the incident initiation database object information system to execute, which in turn causes several database object processes to cause the auto accident incident to be materialized in the database.

A database object is specified to the SQL DBMS through the SQL definition language (DDL). All four components of a database object operate within the “firewall” of the DBMS. This ensures that database objects are protected from improper access or manipulation by 3GLs, or 4GLs. A DBMS that only defines, instantiates, and manipulates two dimensional data structures is merely a simplified functional subset of the DBMS that defines, instantiates, and manipulates database objects

Database objects are completely defined within the database object column of the Knowledge Worker Framework. They are interfaced to the “outside world” by means of business

information systems through SQL views. Each view represents the entire set of data, or some subset of a set of data that truly reflects a known value state of the database object.

Culling out the database objects from 600 or so entities requires three simple questions:

- Does the entity represent only a single value? For example, when the entity, Salary is really a business fact, it should be represented in the Metabase System as a data element

- Does the entity represent a collection of business facts from withing another context? For example, when the entity, Critical Contract Dates, represents multiple business facts, but within the context of the contract, the entity is a property class, and is stored in the Metabase System as such.

- Does the entity represent multiple collections of business facts and is self-contained as to context? For example, when the entity, Contract, contains multiple property classes such as critical dates, signatories to the contract, terms and conditions, items and item quantities, and the like, the entity is a database object and is stored in the Metabase System as such.

The database object diagram is presented in Figure 4 that was created from the Accounts Payable diagram and other finance related ER diagrams. As can be seen, there are only eight database objects. All the other entities from the Accounts payable diagram and from other diagrams associated with finance have been considered and are inferred to be within database objects.

The purpose of the database objects is to present a high level model of the data that embraces the data-scope of what must be accommodated in the Information Systems Plan. Consider the two diagrams. When senior managers meet around a table to discuss database projects, the conversation will certainly be restricted to “big” nouns such as those in the database object diagram rather than the “little” nouns contained in the ER diagram.

During the efforts associating with *executing* the Information Systems Plan, that is, during activities associated with creating specified, implemented, and operational data models, there is more than enough opportunity for the development of third normal form databases.

The database objects are mapped in a many-to-many fashion within the Metabase System to missions. This provides the ability to know which databases support which missions and vice versa.

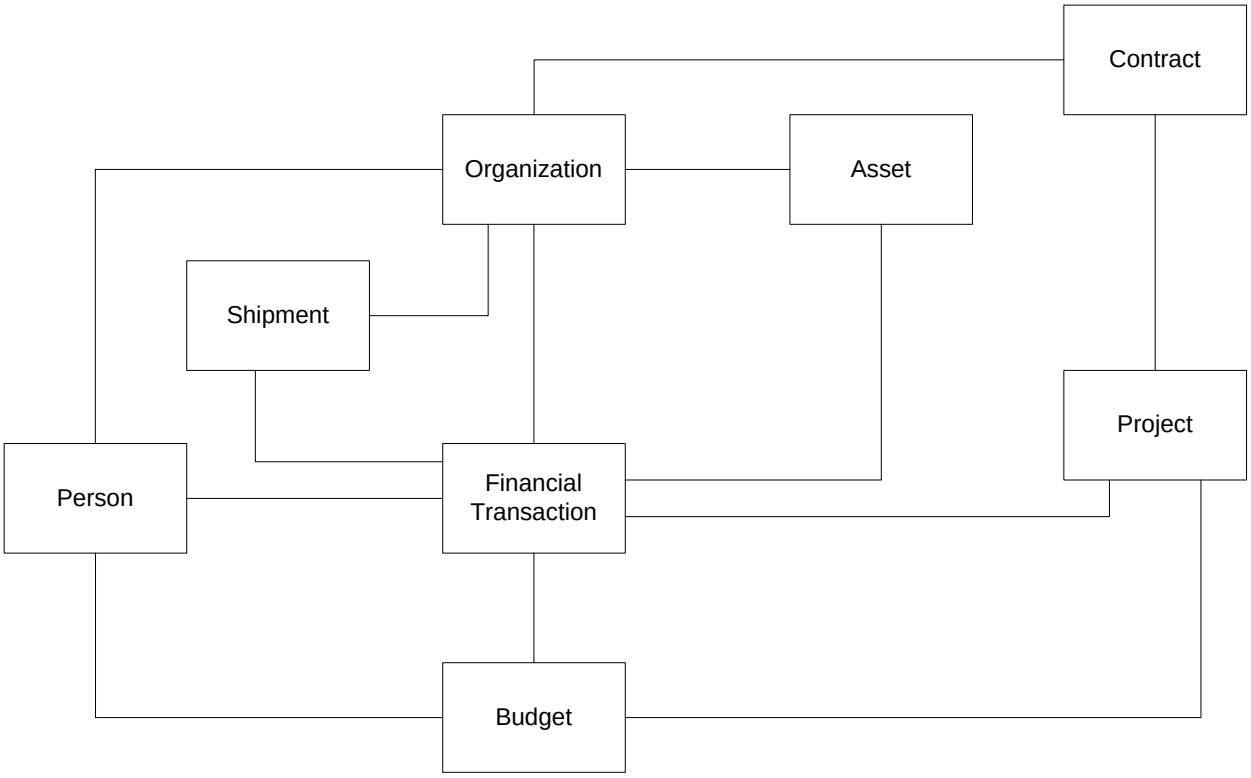

Figure 4. Database Object diagram.

Step 3. Create Resources and the Resource Life Cycles (RLC)[2]

As a short review. Missions are the idealized characterizations of end results of the visionary state of the operating enterprise. Database objects, founded squarely on missions are the high-level declarations of the data required to reflect the achievement of the mission's vision. Resources and their life cycles are the names, descriptions, and life cycles of the critical assets of the enterprise, which when exercised achieve one or more state-based aspects of the missions. Each life cycle is composed of RLC nodes.

[2] The Resource Life Cycle Analysis monograph, authored by Ron Ross and Wanda Michaels was published in 1992 by Database Research Group, Boston Ma. Ron can be reached at www.BSRsolutions.com

A mission might be human resource management, where in, the best and most cost effective staff is determined, acquired and managed. A database object squarely based on human resources would be employee. Within the database object, employee, are all the data structures, procedures, integrity constraints, table and database object procedures necessary to "move" the employee database object through its many policy-determined states. A resource might also be named employee, and would set out for the employee resource the life cycle stages that reflect the employee resource's "journey" through the enterprise. While an enterprise may have 50 to 150 database objects, there are seldom more than 20 resources.

Enterprises build databases and business information systems around the achievement of the life cycle states of its resources. Business information systems execute in support of a particular life cycle stage of a resource (e.g., employee promotion). These information systems cause the databases to change value-state of contained database objects to correctly reflect the resource's changed state. The state of one or more database objects in the database is the proof that the resource's state has been achieved. Resources become the lattice work against which database and business information systems are allocated. The table that follows presents the basic components of resources and their life cycles.

Resources and Resource Life Cycles	
Critical Components	**Definition**
Resource	A resource is an enduring asset with value to the enterprise
Resource Life Cycle	A resource life cycle is the linear identification of the major states that must exist within life of the resource. The life cycle of a resource represents the resource's "cradle to grave" set of state changes.
Precedence Vector	A precedence vector is a relationship between two nodes of different RLCs that indicates that the Target RLC node is enabled in some significant way by the Source RLC node.
RLC Matrix	The RLC matrix is the set of all resources, their life cycles and the precedence vectors among the nodes. Properly drawn the RLC Matrix resembles a PERT chart

Table 8. Resource Life Cycle Components.

The ultimate goal of resource life cycle analysis is the identification and description of the major resources essential to the enterprise's survival, and the ultimate goal of the Information Systems Plan is the identification and accomplishment sequencing of the information systems projects required to implement the enterprise resources in the most effective manner possible.

Step 3.1 Determine the Resources

The enterprise's product and/or service resources are defined; they may be either concrete or abstract. Ron Ross provides two guidelines to assist in resource identification:

- Define the product or service that constitute the enterprise's resources from the customer perspective.
- Define the resource as it is managed between the enterprise and its customers.

In addition, the characteristics of a resource are set out in Table 9.

Characteristic	Definition
Basic	The resource must exist for the enterprise to exist
Complex	The resource requires development and management
Valuable	The resource must be protected, exploited, and/or leveraged by the enterprise.
Enduring	The resource exists beyond business cycles
Shareable	The resource is shared by different functions of the enterprise.
Structured	The resource can be described and organized
Centralized	The resource can be controlled and monitored centrally, even if distributed in creation or use.

Table 9. Resource Characteristics.

Additional tests for resources are:

- The resource must be monitored and forecasted. By the time the resource is required, it is too late to be produced.
- The resource must be optimized. The resource is of such a cost that an unlimited supply is not possible.
- The resource must be controlled and allocated. The resource is desirable and necessary, and must be shared among functions of the enterprise.
- The resource must be tracked. Each stage of the resource is important to the enterprise, including its demise.

Step 3.2 Determine The Resource Life Cycles

The second step is to determine a life cycle for each resource. Each node in the life cycle represents a major state change in the resource. The state change is accomplished by business information systems and is reflected through the enterprise's database objects (conformed into databases). Figures 5, 6, and 7, developed in support of an enterprise database project for a state-wide court information system, shows the resource life cycles for Document, Case, and for Court's Personnel.

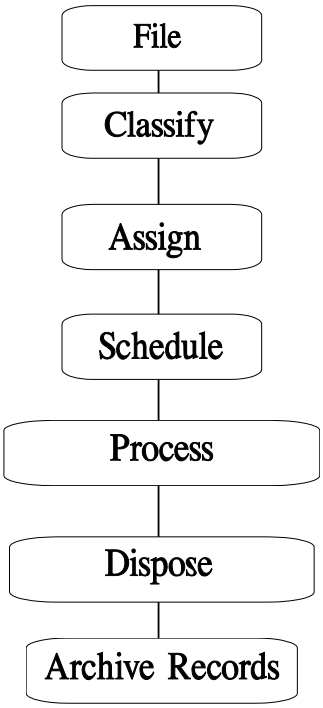

Figure 5. Case.

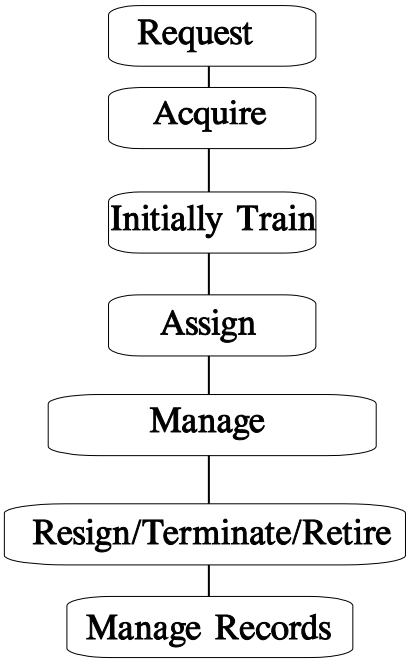

Figure 6. Court Personnel.

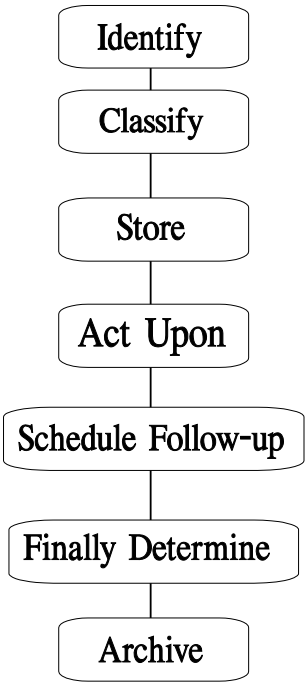

Figure 7. Document.

Step 4. Allocate Precedence Vectors among RLC Nodes

After the resources and life cycles are complete, precedence vectors are established. There are actually two types of precedencies: Within the value chain and between resources. Precedencies *within* the value chain are established during the life cycle analysis. These are the lines that connect one node to the next.

A precedence between resources is created when a resource life cycle state, that is, a specific life cycle node, cannot be effective or correctly done unless the preceding resource life

cycle state has been established or completed. A precedence arrow, renamed precedence vector, is drawn from the enabling resource life cycle state to the enabled resource life cycle state.

The most difficult problem in establishing the precedence is the mind set of the analyst. The life cycle is **not** viewed in **operational** order, but in **enablement** order: that is, what resource life cycle state must exist before the next resource life cycle state is able to occur. This is a difficult mind set to acquire, as there is a natural tendency to view the life cycle in operational order. The test of precedence becomes: what enables what, and what is it enabled by what?

For example, project establishment precedes the award of a contract. This does not seem natural, since a project would not operationally begin until after a contract is awarded. However, there must be an established infrastructure to create the project and to perform the work prior to the contract award. A workforce must be in place to perform work along with the ability to assign work to the employee on the contract, and the ability to bill the customer. Therefore, the project enables the contract.

There are three possible meanings for enablement. That is, a resource life cycle state precedes another resource life cycle state because:

1. The accomplishment of the preceding resource life cycle state saves money.

2. The resource life cycle state leads to rapid development of another resource life cycle state

3. The resource life cycle state permits faster, more convenient accomplishment of another resource life cycle state.

If one or more indicators exists, then a precedence vector should be created.

Two alternatives exist relative to the existence of the enterprise: newly established or existing. Experience shows the preferred perspective is that of an already-existing enterprise.

RLC states may or may not occur during a life cycle, or events may occur in parallel. For example, an employee may receive an award, but then again, may never receive an award. An employee may work before and after a security clearance is granted. The strategy to deal with parallel or optional RLC states is to create a single stream of RLC states in which none are parallel or optional by “pushing down” the parallel or optional RLC states to a lower level.

Figure 8 shows the enable interaction among these three life cycles.

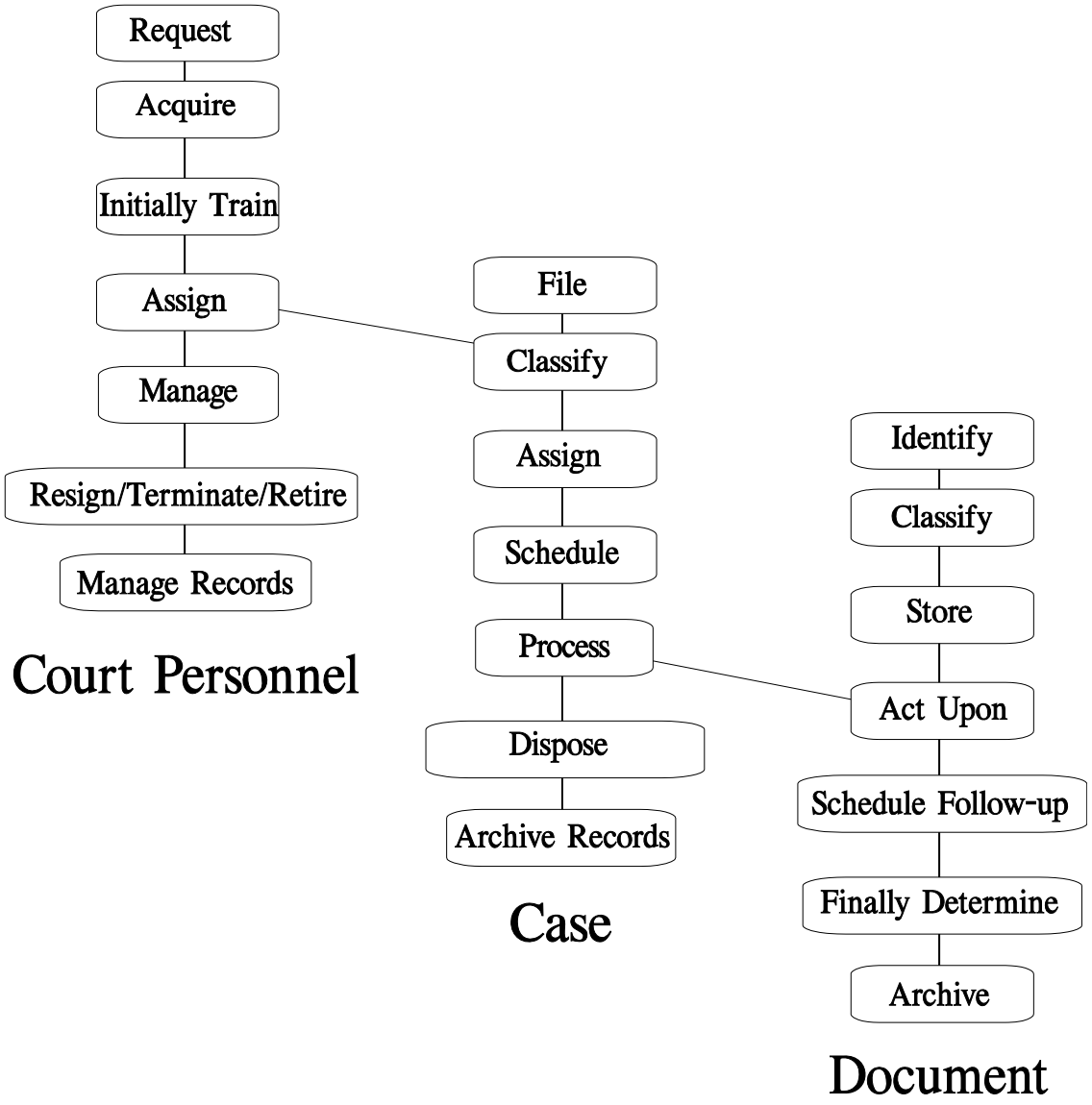

Figure 8. Precedence vectors among Resource Life Cycles.

Step 5. Allocate Business Information Systems and Databases to the RLC Nodes

Once the resource life cycle network has been created, it is stored into the Metabase System. Once there, its lattice can be employed to attach the databases and business information systems. Databases and their business information systems exist within a data architecture framework. The five distinct classes of databases are:

- Original data capture (ODC)
- Transaction data staging area (TDSA)
- Subject area databases (SDB)
- Data warehouses (wholesale and retail (a.k.a. data marts))
- Reference data

Most resource life cycle nodes contain at least one original data capture database application. The data from these ODC databases should be pushed to their respective TDSA databases. Once there, various subject area databases pull the data to build the longitudinal and broad subject area databases. It is likely that there is one subject area database for one or more resources. Data from the subject area databases, also called Operational Data Stores by Bill Inmon, is again pulled to create one or more data warehouse databases. Most databases employ one or more reference data tables as standard semantics for selection, control-breaks and printing.

Databases and business information systems exist in two forms: “as-is” and “to-be.” An “as-is” database or information system, as it’s characteristic implies, represents the existing state of the information technology assets. A “to-be” database or information system is a proposal for some technology improvement, functional enhancement, or an under-way project effort.

Step 5.1 Allocate Existing (As-is) Databases or Files to Resource Life Cycle Nodes

Within the class of existing databases or files, there are three prototypical examples:

- A file for every distinct process or purpose
- A single database for all reasons
- Multi-data architecture database classes

Knowledge about these existing set of databases and files should already reside in the Metabase System. If their metadata is not in the Metabase System, these databases and files must be

discovered. A good way is to research all the reports produced by the information systems department and allocate the file that was employed to produce the report to the RLC node that best fits the representation of the data.

Once all the databases and files are allocated, reports can be produced by the Metabase System that show RLC nodes that have a "bountiful" quantity of databases and files (not a good sign) and those that have no allocated databases or files (also not a good sign). In the later case, there probably are databases and files but they are either "private" or undiscovered. Either case is "not a good sign."

In any case, allocating them to resource life cycle nodes is a matter of distilling the intended purpose of the database or file and then creating the relationship. It is likely that some files or databases will allocate to multiple nodes and even to different nodes of different life cycles. The quality of mapping relationships is inversely proportional to the encapsulation of the data to the resource life cycle node. For ODC databases or files, there should be few multi-node mappings. For data warehouse databases there will probably be many multi-node mappings.

Step 5.2 Allocate Existing (As-Is) Business Information System to Resource Life Cycle Node

Within the class of existing business information systems, there are three prototypical examples:

- Monolithic mainframe with manual subsystems for workflow
- LAN-based, workflow and client/server systems architecture
- Commercial Off-the-shelf software (COTS)

As above, knowledge about these existing set of business information systems should already reside in the Metabase System. If their metadata is not already in the Metabase System, these business information systems must also be discovered. Similar to databases and files, a good way is to again research all the reports produced by the information systems department and allocate the business information system that produced the report to the RLC node that best fits the representation of the data.

Again, similar to databases and files, once all the business information systems are allocated, reports can be produced by the Metabase System to show the RLC nodes that have a "bountiful" quantity of business information systems (not necessarily a good sign) and those that have no allocated business information systems (also not a good sign). In the later case, either there probably are business information systems that are either "private" or undiscovered, or the set of activities necessary to accomplish the work implied by the RLC node is being accomplished manually.

In any case, allocating them to resource life cycle nodes is a matter of distilling the intended purpose of the database or file and then creating the relationship. It is likely that some files or databases will allocate to multiple nodes and even to different nodes of different life

cycles. The quality of mapping relationships is inversely proportional to the encapsulation of the business information system to the resource life cycle node. For ODC databases or files, there should be few multi-node mappings. For data warehouse databases there will probably be many multi-node mappings.

Step 5.3 Allocate Future (To-Be) Databases to Resource Life Cycle Node

Within the class of existing databases or files, there are three prototypical examples:

- A file for every distinct process or purpose transformed to a single database for all reasons
- A single database for all reasons transformed to multiple databases fitting within the five database architecture classes
- A file for every distinct process or purpose transformed to multiple databases fitting within the five database architecture classes

Figure 9 graphically illustrates these three prototypical examples, and then enumerates the factors that must be considered in this analysis.

It is important to notice that none of the options propose the creation of a standard access or ISAM file. This is because all permanent data should be defined as SQL-based files, whether they are single table databases or many hundred-table databases. SQL DBMSs provide many features that greatly benefit persistent data such a standard data definition language, backup and recovery, audit trails, security, ad hoc query language access and update, and sophisticated report writer access.

The metadata for the existing databases or files should already be in the Metabase System as a consequence of Step 5.1. The purpose and scope of the future databases merely has to be defined in terms of names and descriptions of database objects and not any of the data base object details.

Step 5.4 Allocate Future (To-Be) Business Information System to Resource Life Cycle Node

Future business information systems, if properly proposed have three components:

- The existing environment description,
- The future environment description,

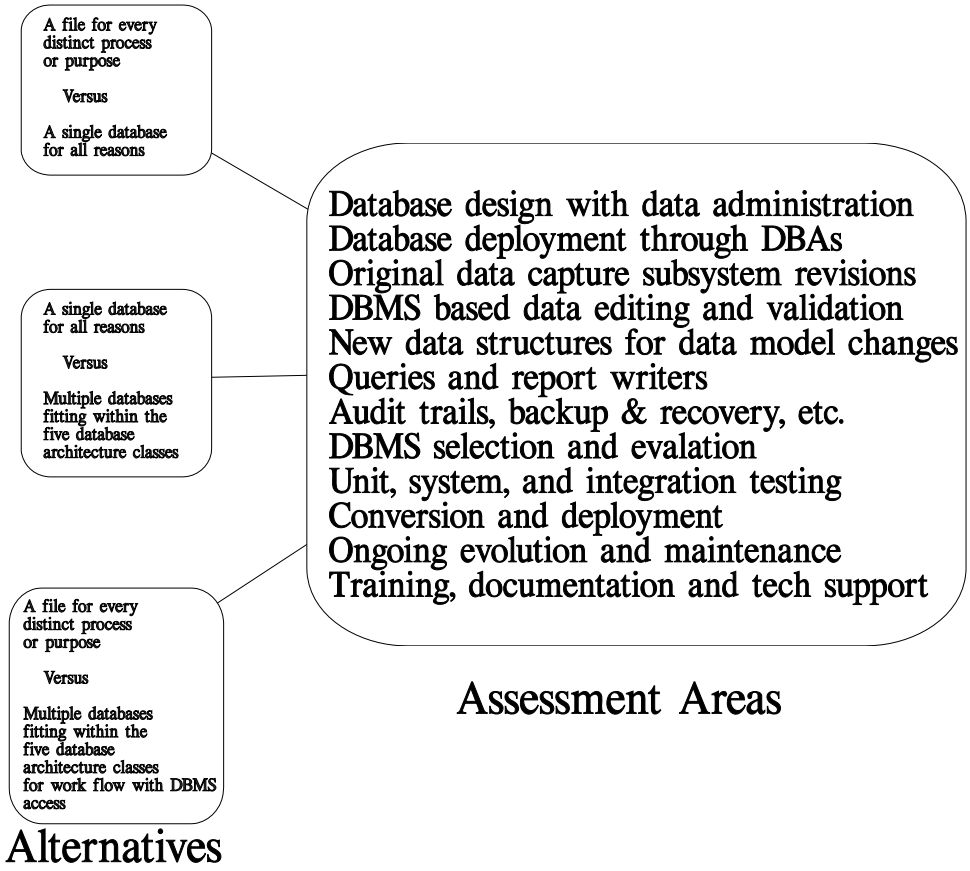

Figure 9. Alternatives and assessment areas necessary to determine preferred project for database/file transformations.

- A high level strategy for accomplishing the transformation from the existing environment to the future environment.

Within the class of future business information systems there are three prototypical examples:

- Monolithic mainframe with manual subsystems with ISAM and COBOL transformed to LAN-based, workflow and client/server
- Monolithic mainframe with manual subsystems transformed to commercial off the shelf (COTS) software

- Monolithic mainframe with manual subsystems with ISAM and COBOL transformed to same but with SQL DBMS access

Figure 10 graphically illustrates these three prototypical examples, and then enumerates the factors that must be considered in this analysis.

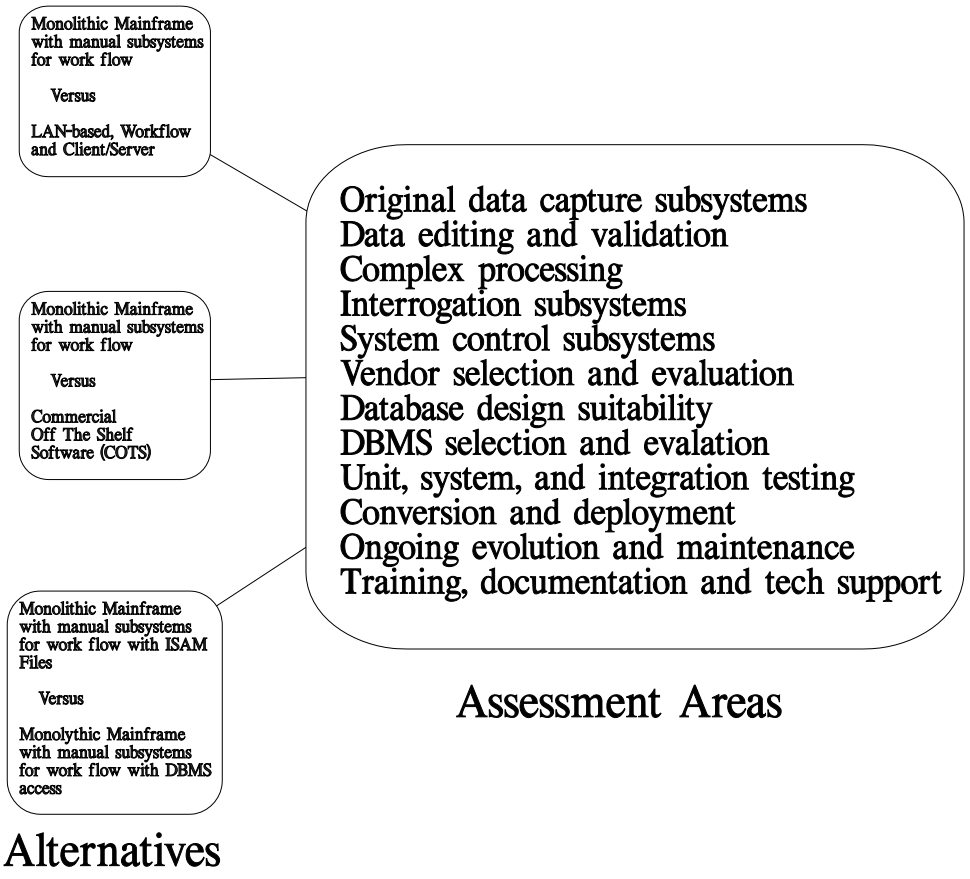

Figure 10. Alternatives and assessment areas necessary to determine preferred project for business information systems.

Any of these alternatives could be enhanced by either Internet or Intranet access. This type of access, if the proper software development environment is employed, is a paradigm shift only if the system is batch. If the system is intended to be on-line through terminals, PCs, or is client/server, the Internet or Intranet should only be a presentation layer shift.

In any case, the high level metadata for the three components that comprise the future business information system must be collected and stored in the Metabase System. The first set of metadata should already be in the Metabase System as a consequence of Step 5.2. In addition, all future business information systems should be cast in terms of future databases.

Step 5.5 Configure Information Systems Plan Projects

Configuring Information Systems Plan projects consists of determining the full set of requirements and then selecting the first cut preferred alternative for carrying out the transformation from the "as-is" environment to the "to-be" environment. The considerations that must be reviewed and addressed are distinct for databases and for business information systems. The figures on the next two pages provide the three prototypical alternatives for each and then the assessment areas that must be addressed.

The final outcome is a full understanding of the proposed future project. This full understanding is then employed in the next step, Allocating Standard Work Breakdown Structures to Each Database and Business Information System Project. Thereafter, each proposed project is input to a project management system and scheduled.

As expected, the items dealing with database or file transformations are primarily centered around identifying the correct set of activities associated with successful database projects. Similarly, the assessment areas regarding business information systems deal mainly with the process, database interface, and presentation layer considerations associated with reading, updating through database objects, and writing data. Each assessment area should be supported by an already existing work breakdown structure and a set of experience based metrics.

A key consideration in any transformation from an existing set of databases and business information systems is the movement of a maximum quantity of functionality to inside the boundary of the DBMS. Whenever a routine is outside the purview of the DBMS, then it must be implemented exactly the same from within every different environment that may affect it.

For example, if data integrity rules are created and executed from within a 3GL COBOL program, then there must absolute assurance that these same data integrity rules are defined and acted upon in exactly the same way if an update is carried out through a 4GL program or a DBMS vendor's query language. Since that would be very difficult indeed, it is wisest for the enterprise to implement data integrity rules from within the domain of the DBMS.

Step 6. Allocate Standard Work Break down Structures (WBS) to Each Business Information Systems and Database Project

The key reason for having a well engineered check list for identifying the types of work involved in either a database or business information system project is the ability to then used canned work breakdown structures (WBS). When these WBSs are coupled with experience-honed metrics that are embedded in a project management system that "self-learns" from on-going projects, accurate, reliable and repeatable project plans result.

Figure 11 presents a very high level view of how project management and the projects associated with RLC nodes are interrelated. In actuality, there are many more tables within the Whitemarsh project management software. But, from this perspective, when the assessment checklist is compiled, the specific WBSs that are applicable are selected from the project template table.

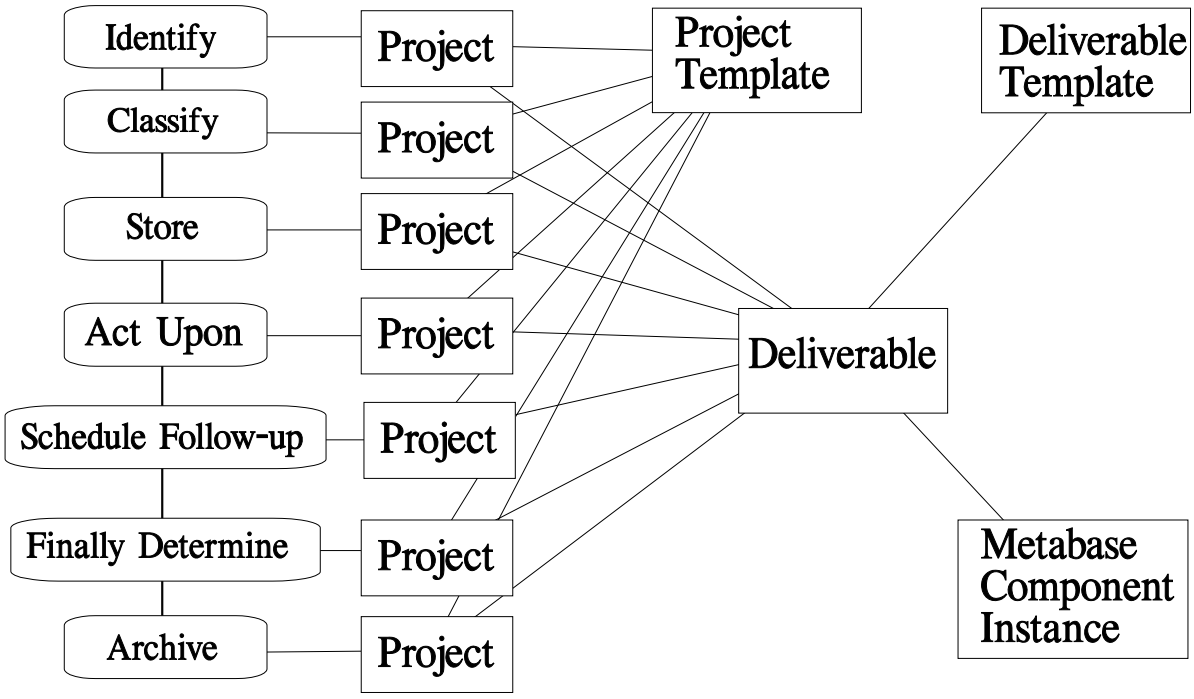

Figure 11. Resource Life Cycle Node project engineering.

The are five distinct classes of projects are:

- Administration and management
- Specification
- Implementation
- Operation and maintenance
- Multiple category

The lists that follow provide the names of all the different projects for which Whitemarsh has standard WBSs and metrics.

Administration and Management

- Administrator Documentation
- DBMS Selection And Evaluation
- Repository Selection And Evaluation
- Enterprise Model Audit
- Information Systems Plan
- Repository Development
- Standard Estimation

Specification

- Conceptual Specification
- Database Process Model
- Detailed Data Model
- Functional Area Specification
- Functional COTS Selection
- Functional Prototype
- High-level Data Model
- Impact on Enterprise Model
- Implementation Strategy
- Mission Model
- System Control Requirements

Implementation

- Functional Area Implementation
- Functional COTS Implementation
- DBMS Schema And View Development

- DBMS Physical Database Specification And Implementation
- Interrogation Development
- System Control Implementation
- System Documentation

Operation and Maintenance

- Emergency Maintenance
- Application Optimization Assessment
- Standard Maintenance
- System Evolution
- Repository Evolution

Multiple Category

- Data Collection And Validation Systems
- Data Conversion
- Training
- Warehouse

Step 7. Load Resources into Each Project

Once the WBS is selected, the WBS list and associated deliverables and metrics are automatically brought into the project. When the quantities for each deliverable type is computed, then the overall gross hours estimate for the project is created.

The gross hours estimate is then finalized (either upwards or downwards) by the selection of work environment factors (e.g., nobody even knows who the users are (that's a bad work environment factor)), and also by the specific persons assigned who have varying levels of capabilities in certain experience levels (e.g., someone is assigned to create the data model who doesn't yet even know the meaning of the term, "ER diagram"). That's a bad staffing factor.

The value in having highly engineered work environment and staffing experience factors that adjust the gross hours is that project managers can then relay back to management the exact reasons why a project will cost more or less than another project of even the same construct and size.

The resources are then exported to a text file that is able to loaded into a project management system such as Microsoft Project. This is necessary because the purpose of the Whitemarsh project management system is to support the planning of projects on an enterprise wide basis rather than the scheduling of individual projects.

Step 8. Schedule through a Project Management Package

Project management systems like Microsoft Project, Welcom's Open Plan Professional, or Primavera's P3e all require PERT (activity network charts) to effectively schedule an entire RLC network of RLC node assigned projects.

When WBSs are brought into a project management system, they are treated as self-contained subprojects within the overall set of RLC node network of projects. Figure 12 shows a RLC network across six different Resource Life Cycles. The resource life cycles are depicted from their first to last node in a top-down fashion. The precedence vectors are shows from one node of a RLC to another node of a different RLC. Multiple precedence vectors do not exist between resource life cycles. When this RLC network is turned on its side, as shown in Figure 13, it resembles a PERT chart. The chart naturally contains parallel sets of nodes that intersect. From this diagram it is easy to see that the network of RLC nodes can be traditionally scheduled.

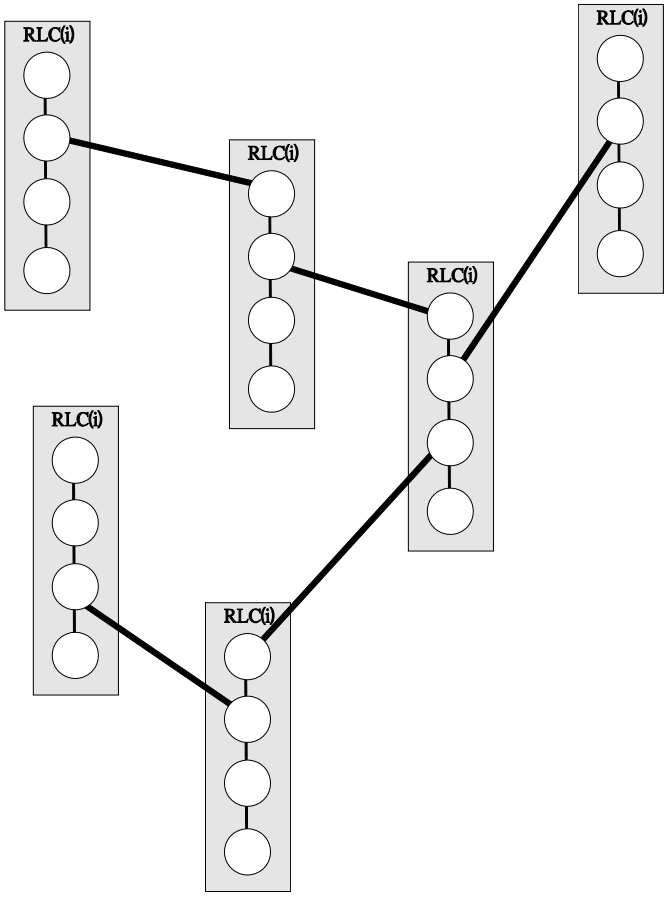

Figure 12. Resource Life Cycle Network.

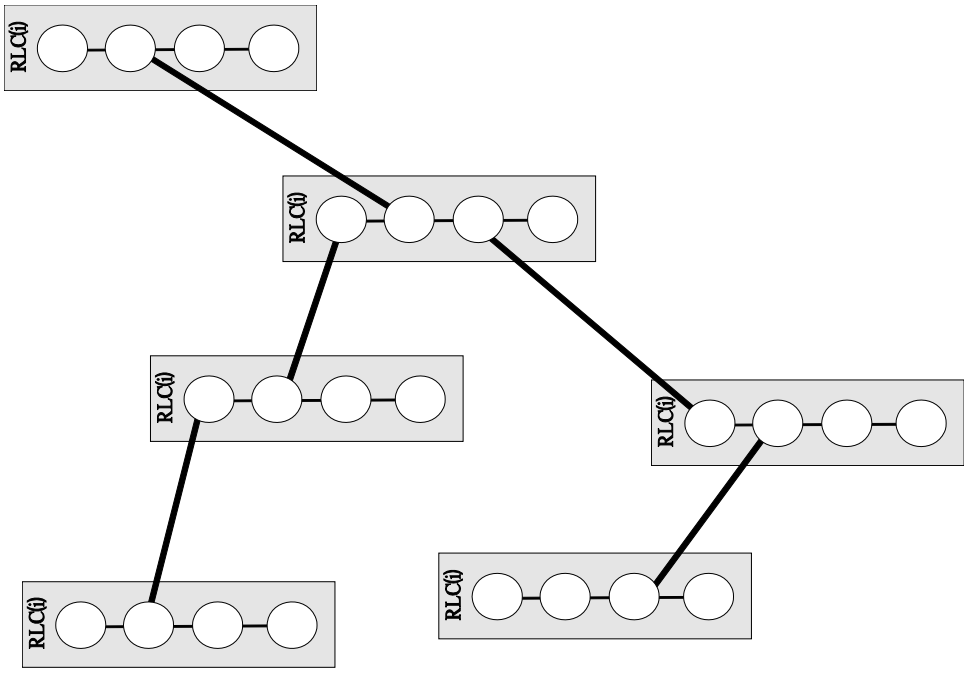

Figure 13. "Pert-like" Resource Life Cycle Network.

Step 9. Produce and Review the Information Systems Plan

When the resource loaded network of projects is scheduled through a project management system, normal results are produced. That is, the enterprise is faced with the requirement for:

- Infinite resources
- Infinite time
- Infinite computer capacity and speed, and
- Zero time allocated by "management" to accomplish all the work.

The Information Systems Plan produced by this technique is thus no more able to be accomplished *on the first pass* than is any other Information Systems Plan. Now, where the Whitemarsh starts to become very different from other Information Systems Plans is that the Whitemarsh Information Systems Plan is fundamentally metadata within a Metabase System database. Because the Whitemarsh Information Systems Plan is "data," it can be reported, queried, updated, recalculated, and reprinted any number of times with only reasonable effort.

A second key distinction is that the data that supports the Information Systems Plan is primarily contributed by and thus supported by management. After all, it is their missions, their

database domains, their database objects, their resources, their resource life cycles, and their precedence vectors between the resource life cycles. The only parts that are truly owned by information technology are the proposals for IT projects that transform an "as-is" database or information system to a "to-be" database or information system.

Since "management" is the source of the information systems projects, as they should be, the key questions that they must answer in order to bring the Information Systems Plan within the boundaries of "mere mortals" is:

- What really needs to be done? (That's expressed as the allocated databases and information systems against the resource life cycle nodes.)
- When is it appropriate to do it? (That's expressed through the enablement vectors.)
- Why does it benefit the enterprise? (That's expressed as the resources and their life cycle nodes.)

In support of answering these questions and thus being able to adjust the resource life cycle network, precedence vectors, resource loadings and the like, the following should be considered:

- Determine essential results versus optional results
- Re-examine and adjust precedence vectors to minimize critical path sequences
- Examine and adjust benefits derived from technology because benefits are soft or can be postponed
- Examine and adjust technology for each system implementation, that is,
    - Old technology will suffice
    - 4GL versus 3GL
    - Code generator versus 4GL/3GL
    - Package versus (4Gl/3GL/Code generators)

These questions can only be answered within a team relationship between business management and the technical staff. As management makes new assertions or adjustments, the Information Systems Plan team can adjust assumptions, WBSs, levels of staff skill, quantities of staff, contractor resources, and the like. As these change, the RLC resource loaded network that has been loaded into the project management system can be rescheduled. Iteratively, a realistic schedule will emerge. One that will not be pleasing to all, but fully justified and understood. It will not be an Information Systems Plan that is based on black magic or that is shrouded in technical mystery. Once the finalized Information Systems Plan is generated, the following work products will have been produced:

- Mission descriptions
- Database domains and entity relationship diagrams
- Database objects
- Resources, life cycles and precedence vectors among the resource life cycles
- Identified and allocated "as-is" and "to-be" databases and business information systems
- Workplans
- Loaded Resources
- Complete Schedules
- Technology choices
- Benefits

While each one of these items has value in its own right, collectively there then exists an Information Systems Plan that is enterprise wide, grounded squarely on the enterprise's missions and resources and is accurately scheduled with experience-honed work plans.

Step 10. Execute and Adjust the Information Systems Plan Through Time

Enterprises, once they evolve beyond their first round of information systems, find themselves transformed from a project and package mentality to a release mentality. Figure 14 illustrates this new continuous flow environment. It is characterized by:

- Multiple, concurrent, but differently scheduled projects against the same subject area database or warehouse database

- Single-database projects that affect multiple subject area and data warehouse databases

- Projects that develop completely new capabilities, that can assess required changes to existing capabilities, and that can accommodate a variety of systems generation alternatives (COTS, package, and custom programming)

The continuous flow environment contains four major sets of activities. The user/client is represented at the top in the small rectangular box. Each of the ellipses represents an activity list to accomplish a specific need. The four basic needs are essentially:

- Need Identification
- Need Assessment
- Design
- Deployment

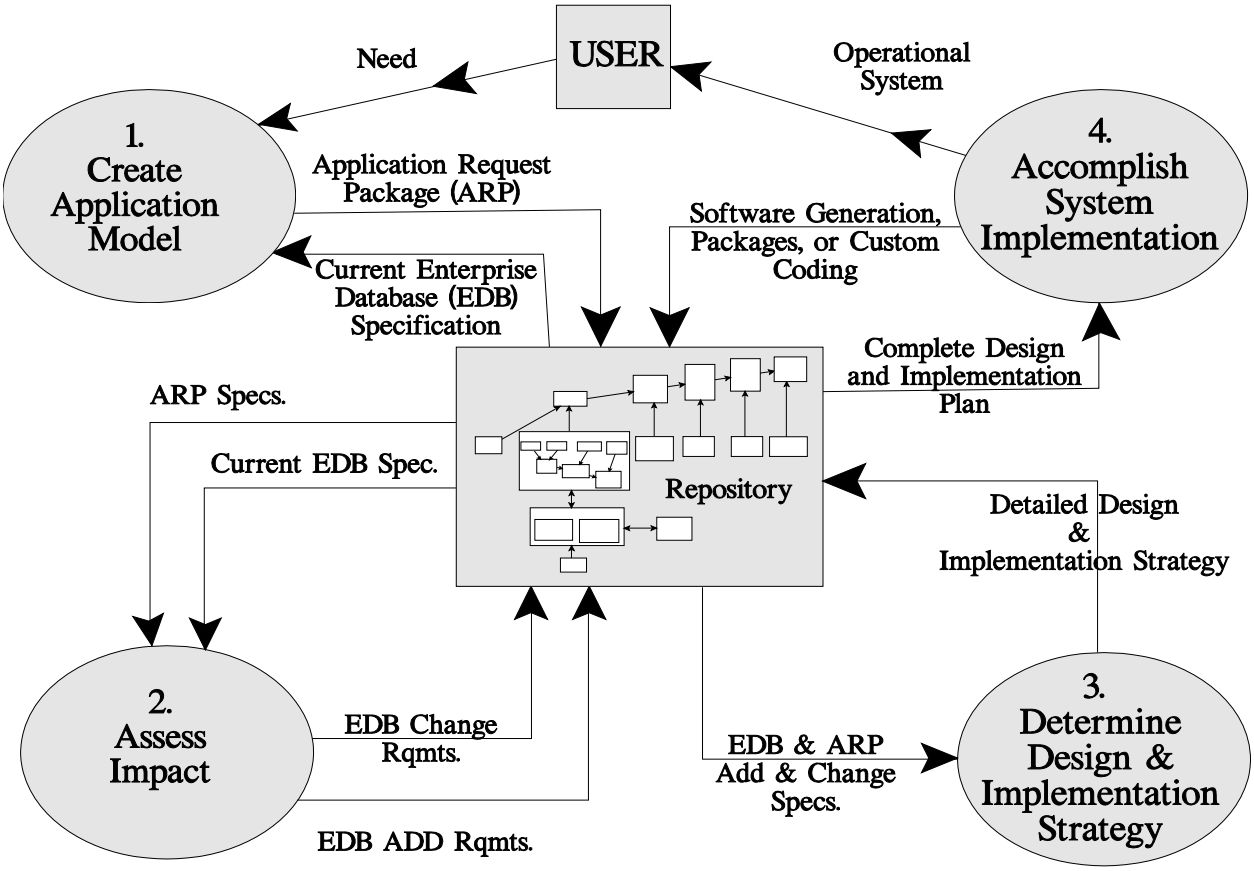

Figure 14. Continuous Flow system development, release-based environment.

The box in the center is the Metabase System. Specification and impact analysis are represented through the left two processes. Implementation design and accomplishment are represented by the right two processes.

In the first process, Create Application Model, the user provides requirements to an application's analyst, who interrogates the existing Metabase System about the existing enterprise database. If the user's request can be handled by existing enterprise features, it is. Otherwise, the application's analyst formulates the requirement into an application request package (ARP). The ARP is represented as a need and includes a series of brief requirement statements in terms of additions to or modifications of an existing enterprise database.

Because this is a continuous-flow model, an ARP may be handled by either existing enterprise database capability, or those now in implementation (process 4), or, in time, through process 3 (design) or process 2 (impact assessment).

At predetermined intervals, for example, monthly or quarterly, all the ARPs are bought into the second process, Assess Impact. During this process, all the ARPs are examined in unison against the then current enterprise database specifications. Because of the continuous-flow nature

of this process, the enterprise database specification may have changed during the time when a set of ARPs is being batched.

When the enterprise database add or change requirements are formulated, they will reflect the state of enterprise database at the time it is to be changed. The form of the enterprise database add and/or change requirements package is similar to that of the ARP. When the impact of a change is assessed, there may also be changes to existing facilities. These changes must be accomplished as well. Thus, when the final enterprise database add or change package is developed, it contains these additional change requirements.

The third process, Determine Design & Implementation Strategy, determines the actual detailed specifications of the computing systems environment changes. The detailed design of the change is in the form of specific changes to existing database schemas, views, computer programs, documentation, procedures, and the like.

In addition to the detailed design of the enterprise database changes, an implementation plan for a project has to be created within the context of the Information Systems Plan, and this individual project plan would include the appropriate PERT chart, WBS, resource loadings, and then a Gantt schedule. Once this project proposal is received and approved, the complete set of changes becomes a new enterprise database version that moves on to the implementation process.

The final process, Accomplish System Implementation, performs all the normal implementation activities. At the end of test and integration, the new enterprise database release is accomplished.

Through this continuous-flow process, several unique features are present:

- All four processes are concurrently executing.
- Changes to enterprise database occur either quarterly or less often and in a very controlled manner.
- The Metabase System always contains all the enterprise database specifications, current or planned. Simply put, if some semantic is not in the enterprise database, it is not corporate information systems policy.
- All changes are planned, schedule, measured, and subject to accounting.
- All documentation of all types is generated from the Metabase System.

5.0 Whitemarsh Information Systems Plan Summary

The summary of this approach to Information Systems Planning is a presentation in Table 10 of its responses to the required characteristics for any Information Systems Plan.

Quality Information Systems Plan Characteristics	
Timely	Creation of the Whitemarsh Information Systems Plan is timely because it can be created in less than three staff years. This is an order of magnitude less than IBM's BSP, Martin's SDP, or Finkelstein's SMP. The time will be even less if its components already exist.
Useable	The Whitemarsh Information Systems Plan enables its users to make both strategic and tactical decisions regarding business information system and database project sequencing based squarely on business fundamentals.
Maintainable	The Whitemarsh Information Systems Plan is maintainable because it mainly uses metadata already essential for Enterprise Database. Metadata that should be readily available in the Metabase System.
High Quality	The Whitemarsh Information Systems Plan is a quality product because it is accomplished through common-sense-based techniques that have been proven over 20+ years.
Reproducible	The Whitemarsh Information Systems Plan is reproducible because at each review, the resources can be re-examined, new technology set into place, basic RLC precedence vectors re-cast, and then the whole plan regenerated.

Table 10. Quality Information Systems Plan Characteristics.

6.0. References

The following references to Whitemarsh materials provide a more detailed exposition practical application of the significant content of this paper.

- Whitemarsh Metabase System software and associated user guides
- Whitemarsh methodology for enterprise database

Paper	URL
Comprehensive Metadata Management	http://www.wiscorp.com/ComprehensiveMetadataManagement.pdf
Metabase Overview	http://www.wiscorp.com/Metabase.zip

Paper	URL
Whitemarsh Data Modeler, Architecture and Concept of Operations	http://www.wiscorp.com/MetabaseDataModelerArchitectueandConceptofOperations.zip
Metabase User Guides	http://www.wiscorp.com/MetabaseUserGuides.zip
Iterations of Database Design	http://www.wiscorp.com/iterations_of_database_design.pdf
Data Management Conferences	http://www.wiscorp.com/dama2002.zip http://www.wiscorp.com/dama2003.zip http://www.wiscorp.com/wrad2000.zip
DAMA 2001 Data Standardization Talk	http://www.wiscorp.com/dama2001datastandardizationtalk.zip
DAMA 2002 Data Standardization, The Problem	http://www.wiscorp.com/dama2003.zip
DAMA 2003 Data Standardization, The Solution	http://www.wiscorp.com/dama2003.zip

The following documents are available for Whitemarsh Website Members. The URLs that follow provide descriptions of the pages. Members should log in and proceed to the appropriate page, e.g., Enterprise Database, find the book, paper, or course and perform the download.

Paper	URL
Data Management Program - Metadata Architecture For Data Sharing Data Management Program - Database Interface Architectures Data Management Program - Projects And Data-Asset Product Specifications Data Management Program - Work Breakdown Structures Knowledge Worker Framework Database Objects Managing Database - Four Critical Factors	http://www.wiscorp.com/wwmembr/mbr_products_edb.html

Paper	URL
Work Breakdown Structures	http://www.wiscorp.com/wwmembr/mbr_products_dp.html
Data Architecture Classes Guidelines for Data Architecture Class - Data Warehouse Iterations of Database Design	http://www.wiscorp.com/wwmembr/mbr_products_dd.html
Work Breakdown Structures Database Project Work plan Templates Information Systems Development Methodology Phases 1 and 2 Whitemarsh Project Estimating Work plan Development	http://www.wiscorp.com/wwmembr/mbr_products_dp.html
Data Management Program - Metadata Architecture For Data Sharing Data Management Program - Database Interface Architectures Data Management Program - Projects And Data-Asset Product Specifications Data Management Program - Work Breakdown Structures Knowledge Worker Framework Database Objects Managing Database - Four Critical Factors	http://www.wiscorp.com/wwmembr/mbr_products_edb.html