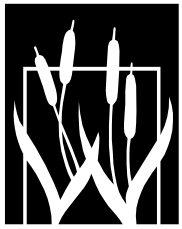

Whitemarsh
Information Systems Corporation

Whitemarsh Project Management: Architecture and Concept of Operations

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

1.0 Project Management in the Knowledge Worker Environment

Whitemarsh project management is designed to help enterprises manage knowledge worker products individually or an enterprise-wide basis. Figure 1 presents the Whitemarsh Knowledge Worker Framework. Any cell from the framework is likely to spawn many projects to engineer, develop, execute, evolve, or evaluate the knowledge worker products implied by the cells.

The examples in this paper are drawn from Information Technology (IT) projects, which naturally are within the Knowledge Worker Framework's "machine" columns. Specifically, IT projects exist within the Systems through Operations rows of the Database Objects and Business Information Systems columns. There is nothing, however, within Whitemarsh project management that restricts its use to these cells or these columns. All projects, regardless of the Knowledge Worker cell from which the project proceeds, have deliverables, task lists, staff assigned, and schedules. And, without proper project management, all knowledge worker projects exhibit the same characteristics: over budget, under specified, delivered late, and failing to meet organizational expectations.

1.1 Why Project Management is Important

Project Management is important because almost all enterprises suffer from one or more of the following problems:

- Inaccurate estimates
- Conflicting priorities among projects
- Inability to deal with varying levels of work conditions, staff skills, and the like
- No intra- and inter-project reporting

Simply put, a common lament is that while there are projects everywhere, the ability to effectively manage these projects on an individual or enterprise-wide basis is nowhere.

For example, studies[1] [2] [3] by have shown that many, if not most, knowledge worker projects exhibit these characteristics: over budget, under specified, delivered late, and fail to

[1] The Standish Group. *CHAOS: 1998: A Summary Review:* 1999. The Standish Group International, Dennis, MA 02638.

[2] Matson, Eric. Speed Kills (the Competition). Fast Company (August 1996). Page 1. Web: www.fastcompany.com/online/04/speed.html.

[3] Strassmann, Paul A. *40 Years of IT History*: 1997. Page 6. Web: www.strassmann.com/pubs/datamation1097/index.html

meet organizational expectations. While not all reasons for failure can be laid at the foot of project management, too many can. Among the underlying reasons are invalid work plans, insufficient time for requirements changes, and inexperienced or mis-allocated staff resources.

Whitemarsh Knowledge Worker Framework

		Man-Machine Interface				
		Machine		Interface	Man	
Deliv-erables	**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	Database object models and specified data models	Information system designs	Invocation protocols, input and output data, and messages	Best practices, quality measures and accomplishment assessments	Job roles, responsibilities, and activity schedules
Tech-nology	Policy execution enforcement	Implemented data models	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
Oper-ations	Operating business	Application interface data models	Operating information systems	Start, stop, and messages	Detailed procedure based instructions	Daily activity executions, and assessments

Figure 1. Knowledge Worker Framework

The United States Government' General Accounting Office (GAO) has been studying IT projects[4] [5] for a number of years, and a review of 10 GAO studies clearly shows that the main reasons why IT systems fail has nothing to do with IT. Again, while not all reasons are specifically related to project management, some of the reasons have to do with critical components of project management. And again, these are invalid work plans, insufficient time for requirements changes, and inexperienced or mis-allocated staff resources.

In response to the need for project planning and management, methods and computer based project management systems have been created. PERT (Program Evaluation Review Technique) was created and employed within the United States Department of Defense in the early 1960s. The PERT chart supplemented the (Henry) Gantt chart and the Critical Path Method (CPM). The PERT chart is typified by a left-to-right network of activities that enforce sequence and precedence. The Gantt chart is a top-to-bottom hierarchical list of activities set against time. A CPM chart is a subset of activities from a PERT chart that takes the longest time. Hence, the critical path. Some Gantt charts simulate PERT charts by serializing some activities. Additionally, once a PERT chart is resource loaded, it can be displayed left-to-right against a time-scale.

In the 1960s through the1970s, project management software operated on large mainframe computers and supported the planning and management of very expensive projects such as Naval ships, bridges, highways, and large IT projects. Because of the project's size, one or more full-time project planners were justified.

With the advent of the PC in the early 1980s, PC packages such has Harvard Project Manager arose. This brought project management to the small project. However, few projects could afford full-time project planners. Because the underlying project planning methods had not changed, and because every project was seen as a one-off effort, PC based project management had become a most dreaded activity.

As a consequence of market pressures and corporate mergers, two classes of project management systems remain today:

- PC based or low-end packages
- Server based or high-end packages

PC based project management systems are typified by Microsoft's Project (www.microsoft.com) or Time Line Solution's product, Time Line (www.tlsolutions.com). Server based project management systems are typified by Primavera (www.primavera.com) and Welcom Software (www.welcom.com).

---

[4] Gorman, Michael M. Knowledge Worker Framework: 1999. Web: www.wiscorp.com

[5] United States Government Accounting Office. Managing Technology: Best Practices Can Improve Performance and Produce Results,: 1997 (GAO/T-AIMD-97-38). Washington, D.C. (web: www.gao.gov)

While the high-end packages are designed for very large, complex project's of thousands of nodes, and while the low-end packages are well suited for scheduling a single project of relatively simple complexity, both the high end and low end solutions do not really address the problems associated with:

- Disjoint projects
- Management of generally uncontrolled resources
- Repeatability of projects
- Incorporation of learned experience into the project estimation cycle

Many knowledge worker projects involve persons from within different organizations over whose time the project manager may not have direct control. Thus, the best the project manager can do is to request participation and to create approximate schedules that show deliverables from these non-controlled participants.

If the knowledge worker project manager creates elaborate project schedules based on many layers of intricately crafted activity networks, then while they look magnificent the instant they are first created, these project plans cannot withstand assaults from all the schedule conflicts. Once these assaults are underway, the project manager has to continuously adjust the layers of project activity networks, resource estimates, parallel and serial paths, etc. Soon the project manager's life is consumed by project management rather than project accomplishment. The dilemma then becomes:

- Accomplish the project, or
- Plan the project's accomplishment.

All too often, project planning is discarded because the project management system, initially thought to be the savior from chaos actually had become another source of chaos. The castle of project management becomes the project manager's dungeon wherein time is the dungeon master, the PERT chart is the shackles, and the schedule is the rack.

To be successful at Knowledge Worker project management, an approach must:

- Concurrently manage disjoint projects
- Manage generally uncontrolled resources
- Enable maximum re-use of past efforts
- Incorporate learned experiences
- Not require a full-time project planner
- Support what-if resource allocation scenarios
- Enable management to know about and view all projects and resources across the enterprise
- Support the presentation of projects individually, or from the perspective of a business-defined set of priorities

1.2 Continuous Flow Process

IT projects are accomplished within distinct development environments. The two most common are: discrete project and release. The discrete project environment is typified by completely encapsulated projects accomplished through a water-fall methodology.

In release environments, there are a number of different projects underway by different organizations and staff of varying skill levels over which project managers may not exert draconian control. Once a large number of projects are underway, the ability of the enterprise to know about and manage all the different projects degrades rapidly. That is because the project management environment has been transformed from discrete encapsulated projects into a continuous flow process of product or functionality improvements that are released on a set time schedule. Figure 2 illustrates the continuous flow process environment that supports releases. The continuous flow process environment is characterized by:

- Multiple, concurrent, but differently scheduled projects against the same enterprise resource[6]
- Single projects that affect multiple enterprise resources
- Projects that develop completely new capabilities, or changes to existing capabilities within enterprise resources

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

- Need Identification
- Need Assessment
- Design
- Deployment

The box in the center is the metabase. Specification and impact analysis is represented through the left two processes. Implementation design and accomplishment is represented by the right two processes. Two key characteristics should be immediately apparent. First, unlike the water-fall approach, the activities do not flow one to the other. They are disjoint. In fact, they may be done by different teams, on different time schedules, and involve different quantities of products

[6] An enterprise resource represents an essential component of the business. Resources provide the business context for projects. For example, staff, money, contracts, equipment, and facilities. More on resources, resource life cycles and resource life cycle networks is found in the Whitemarsh Knowledge Worker and Information Systems planning books, papers, and courses.

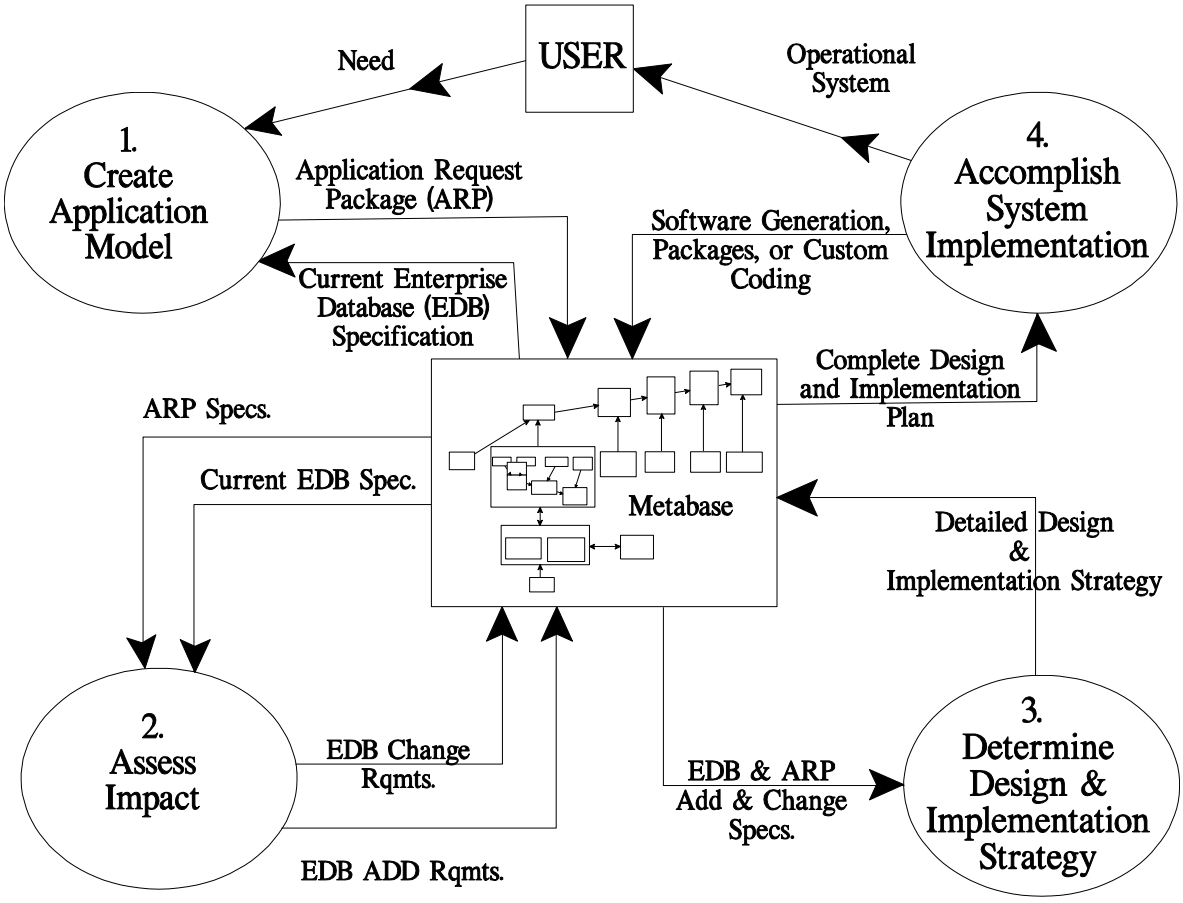

Figure 2. Continuous Flow Process

under management. In short, these four activities are independent one from the other. Their only interdependence is through the metabase.

The second characteristic flows from the first. Because these four activities are independent one from the other, the enterprise evolves by means of releases rather than through whole systems. If it evolved through whole systems, then the four activities would be connected either in a waterfall or a spiral approach, and the enterprise would be evolving through major upgrades to encapsulated functionality within specific business resources. In contrast, the release approach causes coordinated sets of changes to multiple business resources to be placed into production. This causes simultaneous, enterprise-wide capability upgrades across multiple business resources.

In the first process, Create Application Model, the user provides requirements to an application's analyst, who interrogates the existing metabase about the existing enterprise resource. 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

application request package 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 application request package may be handled by either an existing enterprise resource capability, or those 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 application request packages are bought into the second process, Assess Impact. During this process, all the application request packages are examined in unison against the then current enterprise resource specifications. Because of the continuous-flow nature of this process, the enterprise resource specification may have changed during the time when a set of application request packages is being batched. When the enterprise resource add or change requirements are formulated, they reflect the state of the enterprise resources at the time it is to be changed. The form of an enterprise resource add and/or change requirements package is similar to that of the application request package. 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 resource 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 systems changes. The detailed design of the change is in the form of specific changes to existing enterprise systems (manual, automated, or mixed).

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

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

- All four processes are concurrently executing.
- Changes to enterprise resources occur in unison, periodically, and in a very controlled manner.
- The metabase always contains all the enterprise resource specifications: current or planned. Simply put, if an enterprise resource semantic is not within the metabase, it is not enterprise policy.
- All changes are planned, scheduled, measured, and subject to auditing, accounting, and traceability.
- All documentation of all types is generated from the metabase.

1.3 Relationship to Other Metabase Components

Figure 3 presents a high level diagram of the domain of the metabase. As illustrated by the diagram, persons, through their role within an organization perform functions in the accomplishment of enterprise missions, 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. Prior to computers, business information systems and databases were entirely manual. Today, most, if not all are automated in some manner.

Knowledge work is accomplished in support of the enterprise's missions. Projects cause knowledge work to be accomplished. Some enterprises accomplish all knowledge work through projects. Other enterprises employ projects only to establish or change knowledge work environments. In either case, Whitemarsh project management can greatly enhance the ability to have efficient, reliable, and repeatable projects.

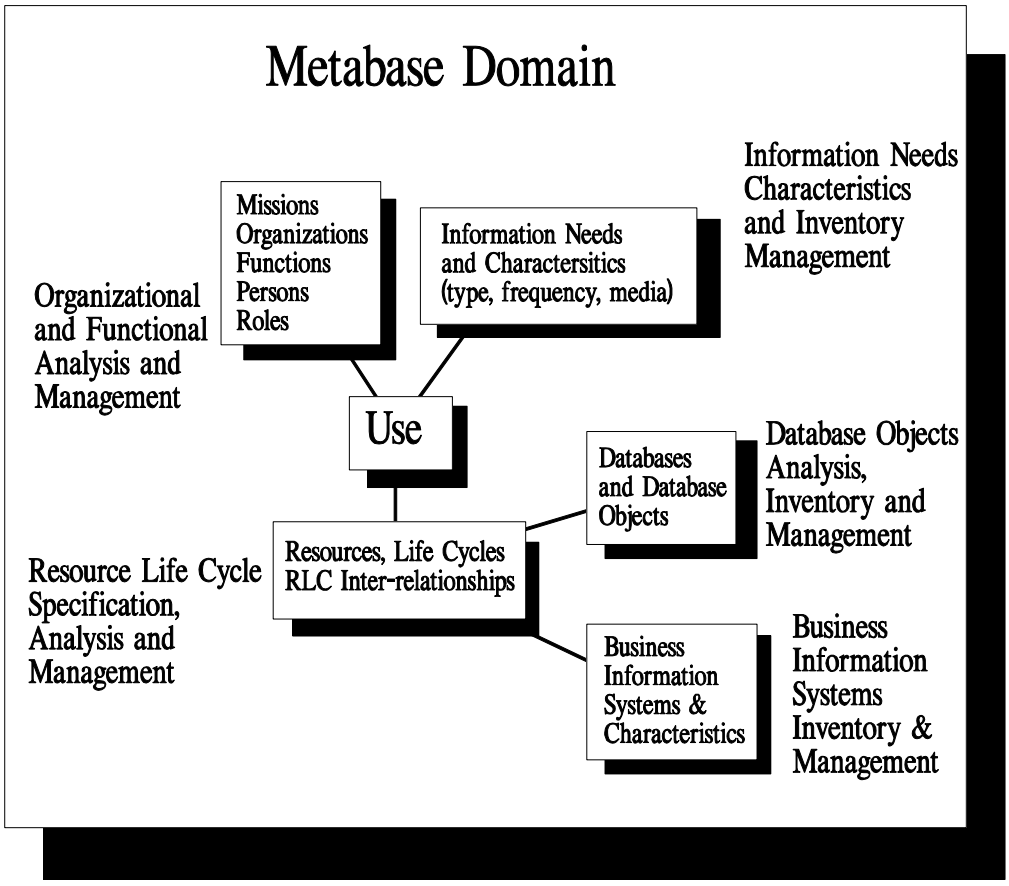

Figure 3. Overall Metabase Environment

1.4 Whitemarsh Project Management Environment

Whitemarsh project management is based first and foremost on its database design. The general "life cycle" of Whitemarsh project management is:

- Employ project, deliverable, and task templates to plan projects
- Plan and estimate projects in a gross way and accommodate different work environment factors
- Staff projects and generate schedules
- Record progress towards deliverable accomplishment
- Re-plan projects as needed
- "Learn" from actual durations from accomplished deliverables

Whitemarsh project management does not, however, support the creation of:

- Very precise parallel and serial networking of projects or tasks
- Very detailed and precise scheduling
- Gantt, PERT or CPM diagram production

These three activities are the proper activities of both low-end and high-end project management systems. In support of these systems, the Whitemarsh project management system generates output data files that can be input into these systems. These systems can then be used to create very precise schedules, activity diagrams and the like. Whitemarsh holds however, the proposition that if accomplishing these three activities were THE basis for successful project management then there would be no need for Whitemarsh project management. While very precise schedules, activity diagrams and the like are important, it seems clear that these features have little or nothing to do with project management success.

In contrast, Whitemarsh believes that project management success is predicated on different activities, which are:

- Continuous optimization of repeatable projects,
- Accommodation of various work environments and factors within these environments,
- Adjustment of project schedules based on differing staff and skill levels, and

- Capturing actual work accomplishment metrics that support earned value analysis and reporting.

The project management database design employed by Whitemarsh has been implemented several times as the basis for project management over the past 10 years. Whitemarsh believes therefore that the "design bugs" are worked out. Whitemarsh project management serves the need of the independent project manager who has to accomplish the definition, management and reporting of diverse and possibly disjoint projects with staff of varying skill levels within mixed work environments that are generally not within direct control. Whitemarsh believes that this type of knowledge worker environment is the rule, not the exception.

2.0 Whitemarsh Project Management, A Difference in Kind

A key difference between the Whitemarsh project management approach and others is that the Whitemarsh approach concentrates on the management of "nouns" while other project management approaches focus on the management of "verbs." Clearly, since there is no one sacred, perfect way to produce a deliverable (i.e., the nouns), if the focus of project management is to identify and control the "methods" (i.e., the verbs) by which deliverables are produced, then to have enterprise-wide project management and/or to have enterprise-wide metrics, the enterprise must first carve-into-stone the processes by which work is done. Not only is this impossible, it is highly undesirable. It is impossible because it is inconceivable that there is only one way to accomplish any product. It is undesirable because it is insulting to project staffs to presume to control their every technique, process and step. Not only can't it be done, no one will allow it to be done.

In contrast to managing "verbs," Whitemarsh project management manages "nouns." It does this by collecting the quantities of resources expended to produce deliverables. Whitemarsh project estimates are therefore based on the staff hours required to produce deliverables rather than to accomplish tasks.

This technique enables different styles of project management to be employed or be set one against the other by comparing the resources expended to produce deliverables. There might be one project template for mainframe development, another for micros, and finally a methodology for web-based systems even though all the deliverables might be essentially the same. Alternatively, there might be multiple project templates that produce the same set of deliverables to serve the needs of different styles or techniques as might be the case for the data-driven and process driven approaches.

Additionally, the Whitemarsh project management approach enables enterprise-wide project reporting in terms of the cost and effort to produce deliverables versus the accomplishment of activities. As work techniques improve, either through the increased skill of staff, or through the adoption of different techniques, the efforts remain comparable because it is the quantity of resources expended to produce the deliverables that are compared rather than the activities, which are no longer able to be compared because they are now different, that produce the deliverables.

To illustrate, when you go into a grocery store and buy an apple, the cost is expressed in terms of the product you are buying, the apple. While you may wonder how much the various activities cost that ultimately produced the apple, fundamentally, you probably do not care. When you go to five different stores and compare the cost of apples (given a standard for equating quality), again you are only comparing the cost of the deliverable, the apple. If one store spends 10% for transportation and another spends 8%, you probably don't care. It's the final cost of the apple that matters, nothing else. So also should it be with project management. The only thing that should matter is the final cost of the deliverable. Nothing more, and nothing less.

If however you are a wholesale apple buyer that deals with a co-operative and by contract, you have to pay every apple grower the maximum cost incurred by any one member of the

cooperative, then you have a real incentive to look "behind" the costs of the deliverables (the apples) to find the different underlying processes that make the costs different. Even then, the goal then is to find the lowest-cost set of activities, and to then highly recommend that set of activities to all members of the cooperative so that your costs for the deliverable–as a buyer–will go down. So, while there may be an interest in activity-sets, they are not the driving force. So too with Whitemarsh project management wherein the cost of deliverables rather than the cost of methods is the driving force.

Whitemarsh project management enables the melding project templates with selected:

- Task templates–that is, the enterprise's techniques, methods or work breakdown structures that have been proven of the years to accomplish in work the most cost effective manner.

- Deliverable templates–that is, the enterprise's specifications of and unit effort metrics required to accomplish the components of its Knowledge Worker products.

The resulting Project Templates are then specially tuned into "real" projects by determining the quantity deliverables, and then affecting the resulting "norm" estimates through:

- Work environment factors–that is, the effects from varied work environments on the creation of deliverables according to certain task templates.

- Staff–that is, the effects from persons and their varying types and degrees of skills on the rate of production of deliverables according to the task templates.

Collectively, these four project management components are an exemplary use of the database fundamental, *define once, use many times*. Whitemarsh believes it has achieved the ability to have maximum reuse with minimum original, one-off effort.

3.0 Architecture and Concept of Operations

Whitemarsh Project Management is squarely founded on a database application that captures and manages the data critical to effective project management. The database's design is depicted in Figure 4, and consists of a number of entities. All these entities are traditional and are interconnected through one-to-many relationships except for those entities that show a one-to-many relationship from the entity to itself. Organization (upper right) contains such a relationship. This relationship means that the entity contains subordinate organizations. For example, an Information Technology organization contains the Information Resource Management organization, which in turn may contain the Data Administration organization, and Database Administration organization. The eight entities that are recursively related are:

- Contract
- Deliverable
- Deliverable Template
- Organization
- Project Template Type
- Resource
- Task
- Task Template

The entities from Figure 4 are also divided into six distinct clusters, which are:

- Contracts, organizations and contract [staff] resources
- Resources
- Project, Deliverable, and Task Templates
- Project Staff
- Project Building and Estimation
- Project Work

In general, the *Contracts, Organizations, and Contract [staff] Resource* cluster of entities represent the environment within which projects take place.

The *Resources*[7] entity represents the target of the project, that is, the area of the business benefitted by the project. For example, for manufacturing, finance, human resources, or land use planning.

[7] Resource has the exact same definition as does *resource* within the Whitemarsh metabase. Please refer to the Whitemarsh Glossary and/or to Resource and Resource Life Cycle Analysis for these concepts.

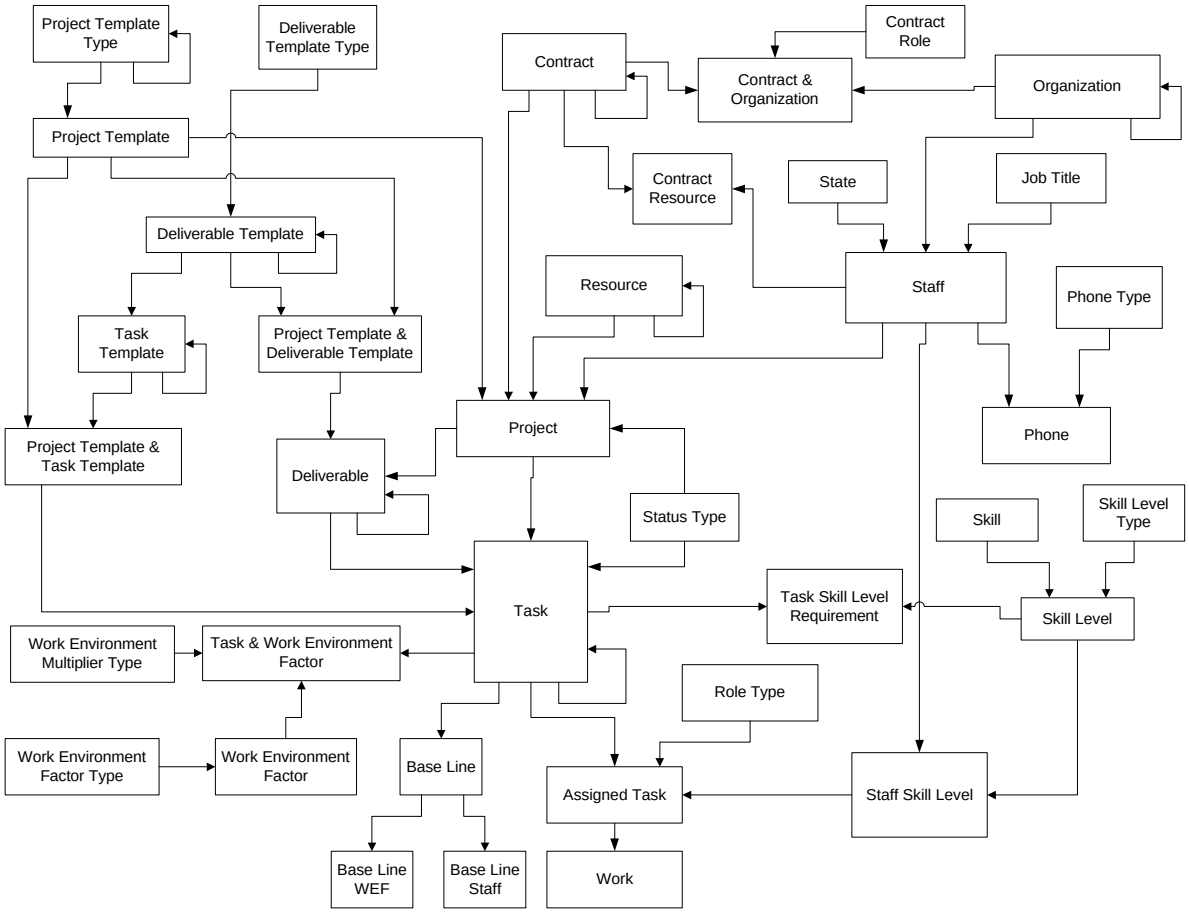

Figure 4. Overall Database Architecture for Whitemarsh Project Management

The *Projects, Deliverable, and Task Templates* entity cluster enables the definition of the templates employed in the actual building of projects. Defined across the enterprise, these templates enable standard project execution and accomplishment measurement.

The *Project Staff* entity cluster enables the inclusion of the staff as resources for a contract, and also permit the specification of the specific types and performance ratings of skills that a person may bring to a specific project.

The *Project Building and Estimation* entity cluster represents the entities that support building projects. Projects and associated tasks are initially created through the use of the Project Deliverables, and Tasks Templates. Once projects and associated tasks are created, they are modified by attaching work environment factors and specific skill-level based staff assignments. Only then can task and project resources be computed.

Finally, as task work is accomplished, the *Project Work* entity is valued. As actual work is accomplished, it can be reported through any of its related entities.

Because Whitemarsh project management system is implemented as a database application, it supports the following types of reports:

- Projects and project statistics of a certain project template
- Projects and project statistics within certain [business area] resources
- Projects and project statistics by deliverable types
- Projects and project statistics by organizational units
- Projects and project statistics by specific project staff members
- Projects and project statistics by certain types of skills
- Projects and project statistics according to certain status types
- Projects and project statistics according to certain work environment factors

The remainder of this section presents the architecture and concept of operations of Whitemarsh project management.

3.1 Contracts, Organizations, and Contract [staff] Resources

Figure 5 has the organization, contract, organization & contract, contract role, and contract resource entities grayed. The organization entity is recursive and represents the various organizations participating in one or more contracts. The contract entity is also recursive. This enables subcontracts. The intersection entity, Contract & Organization, is only allowed to exist if it relates contracts and organizations that themselves do not have subordinate contracts or organizations. This “leaf-only” tagging is enforced by not allowing an organization or a contract record to be tagged unless the record is at the bottom of its respective hierarchy. Once tagged, and after the intersection entity instance is built, there exists a many-to-many relationship between organization and contract. The role played by the various organizations on the contract

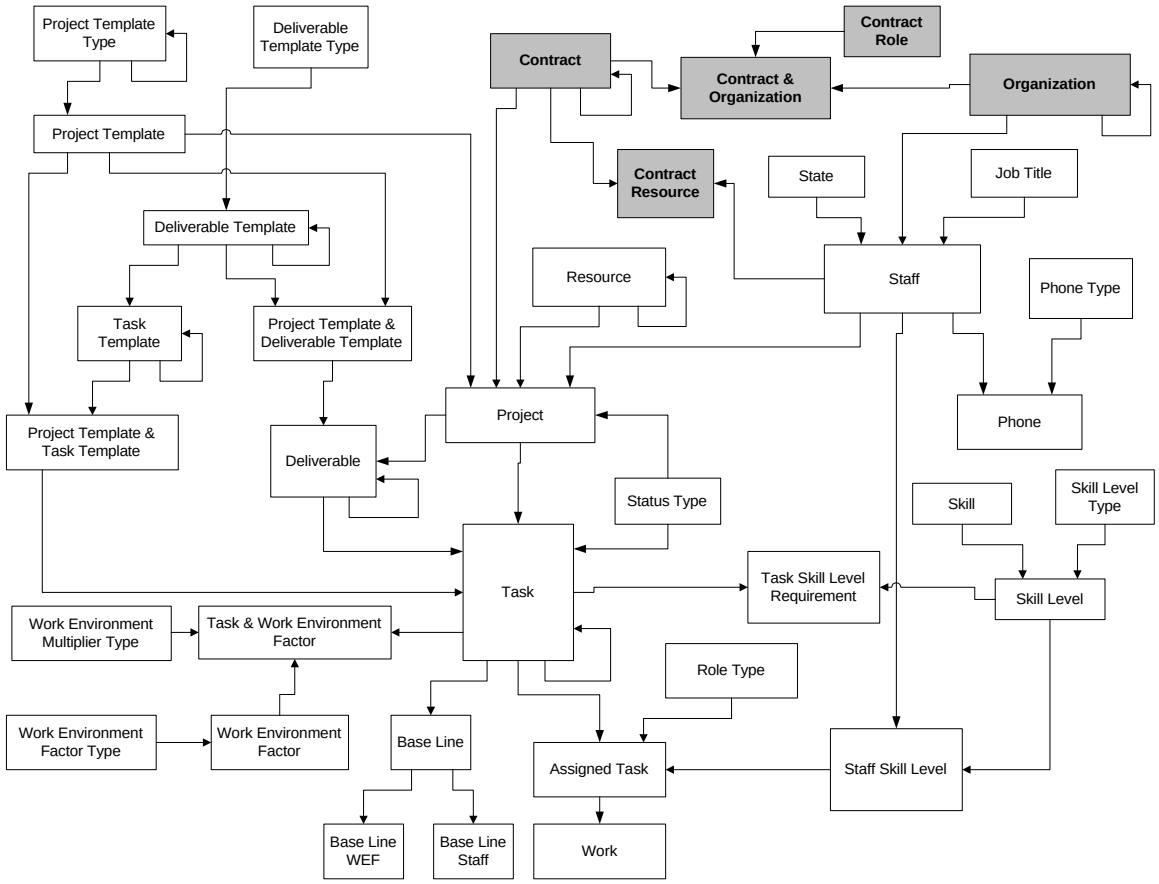

Figure 5. Contracts, Organization, Contract [Staff] Resources

is identified by the contract role entity. Contract roles might include: contract owner and contract participant. More specific roles are identified within assigned tasks through Role Type.

A contract represents the overall working agreement that may govern a collection of subordinate contacts and related projects. In addition to encompassing projects, each contract has a relationship between it and contract resources. Staff may be identified from different organizations as contract resources. The staff to project relationship serves to simply identify the project's leader. Staff work assignments are made through the relationship between Staff Skill Level and Assigned Task.

The relationship is between organization and staff and is used to indicate the organization for whom the project staff member works or is organizationally allied.

In the case of consulting staff, that is those employed by other corporations, a choice has to be made as to whether to list the consultant's employment organization or the organization within the enterprise for whom the consultant is performing the work. To this question, the following suggestion is offered. Make the consultant's organization the organization for whom

they are performing the work if the "customer" of the project perceives that the consultant is working for the "inside" organization. If the "customer" is directly dealing with the consulting organization then the consultant's employment organization should be chosen.

3.2 Resources

Figure 6 has the resource entity grayed. This entity is recursive and represents the hierarchical breakdown of resources of the business. For example, if one of the business's resources is information technology, and within the information technology resource there are information systems, there may be different information system classifications such as Finance, Human Resource Development, Distribution, and Manufacturing. And within Finance, there could be accounts payable, accounts receivable, asset management payroll, and general ledger.

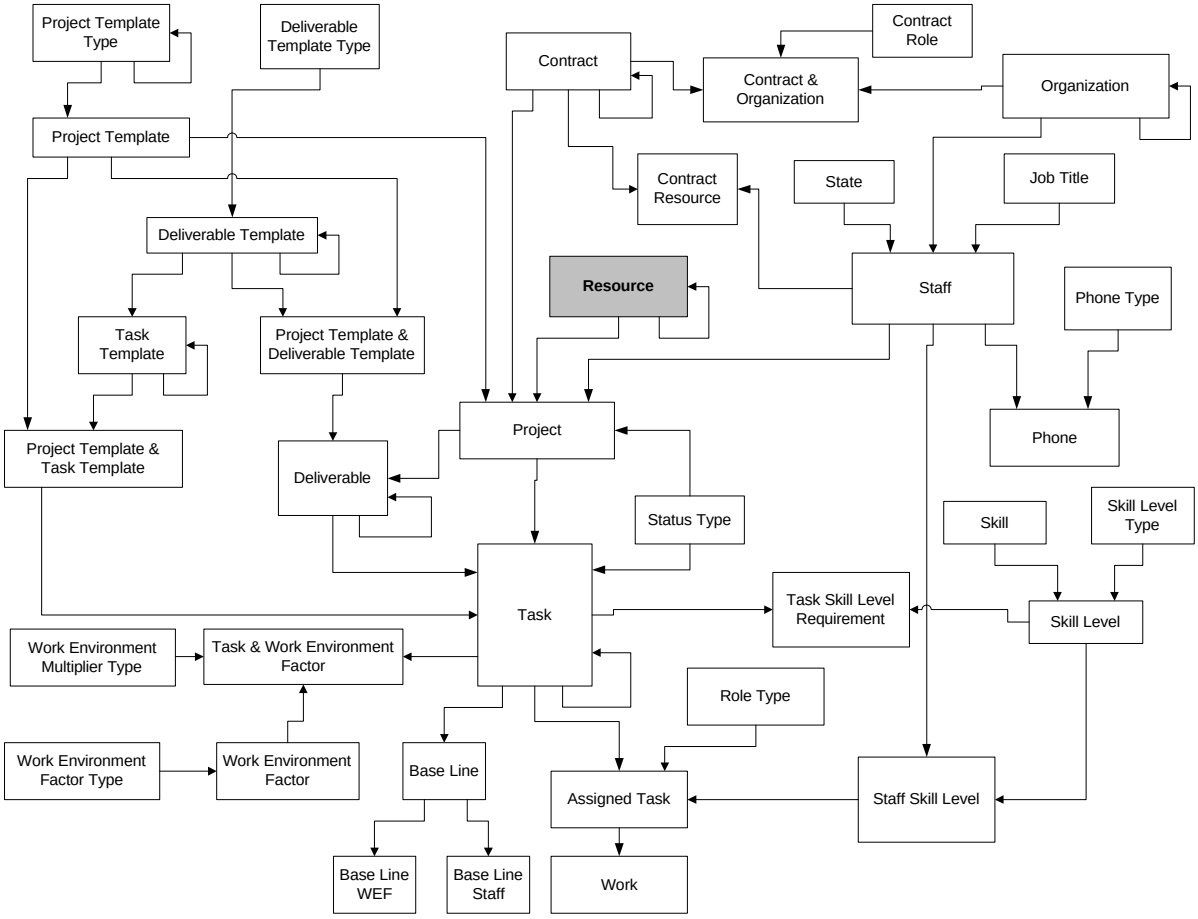

Figure 6. Resources

Another resource might be human resources. Projects addressing that business area could deal with policies and procedures, recruitment, succession planning, in-service education, and the like. Within in-service education could be projects that address in-service education needs identification and assessment, planning, development, execution, and evaluation.

The resource entity is the same as implemented within the Whitemarsh metabase. It provides the lattice work on which to attach the various projects that accomplish improvements or changes within an enterprise's resource. Whitemarsh project management provides the details of these projects. More information about resources, resource life cycles, and networks is available from within other Whitemarsh materials.

3.3 Project and Deliverable Templates

Figure 7 has the following entities grayed:

- Deliverable Template
- Deliverable Template Type
- Project Template
- Project Template and Deliverable Template
- Project Template and Task Template
- Project Template Type
- Task Template

The purpose of these entities is to pre-define classes of projects that are accomplished one or more times. Even when a project is expected to be accomplished only once, it must first be defined as a project template. The reason is simple: No matter how many times you may protest to the contrary, a project is never done only once. Further, the effort to create a project from the template requires the push of only one button. Hence there is no real cost, only a real benefit.

A Project Template can be seen as being within a hierarchical class of projects. This gives rise to the notion of project template types. While clearly the strategy for identifying hierarchical classes of project template types is subjective, the following project template types are illustrative:

1 Training
1.01 Needs Assessment
1.02 Development
1.03 Execution
1.04 Evaluation
2 Information Technology
2.01 Needs Assessment
2.02 Tool Development

2.03 Component
2.03.01 Requirements
2.03.02 Specification
2.03.03 Implementation
2.03.04 Evolution
3 Metabase
3.01 Requirements
3.02 Component
3.02.01 Development

These classes of project templates may mirror the various sections of a mission statement and may take on several generic prefix “forms” such as: plan..., execute...., and evaluate....

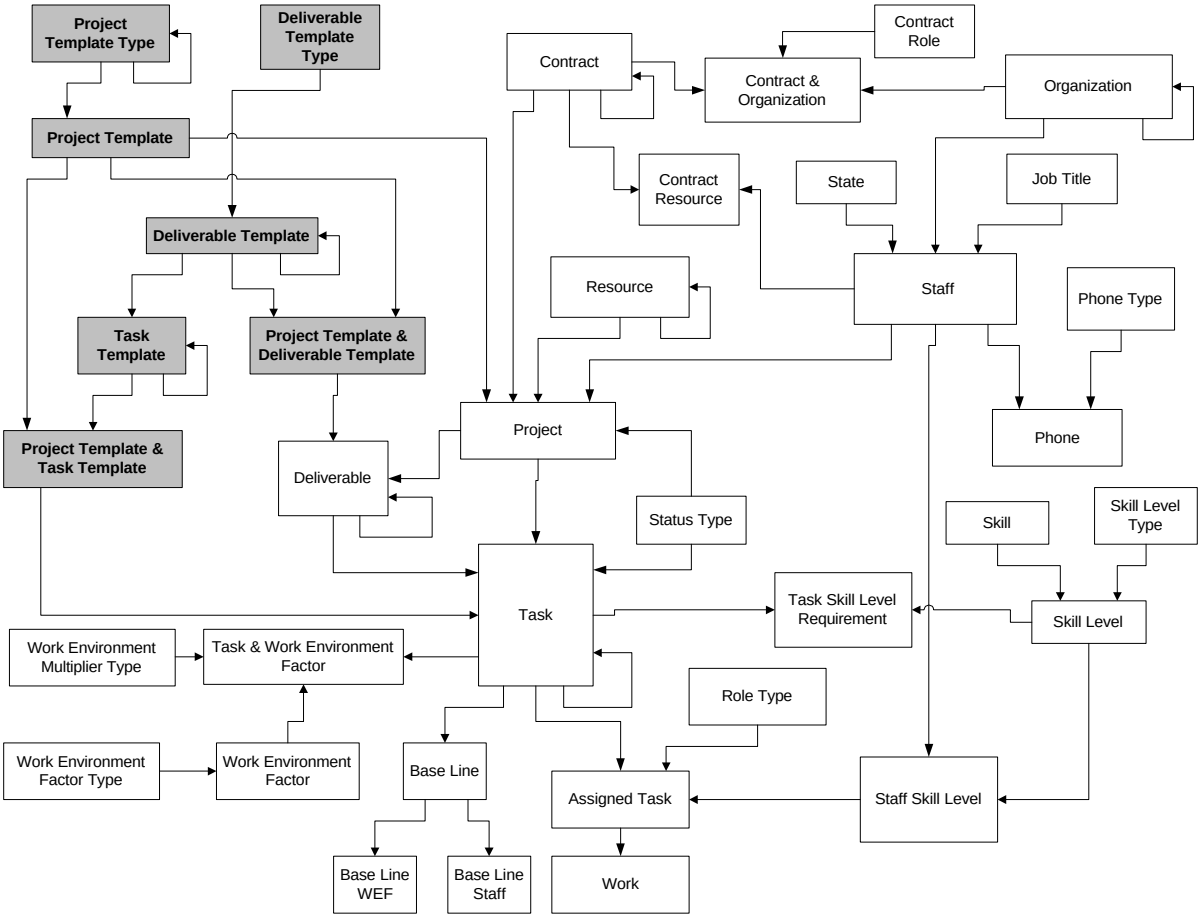

Figure 7. Project, Deliverable and Task Templates

Once project template classes are identified, the individual project templates within each class can be identified. For example, the following project templates would be suitable for the project template type, Metabase Component Development.

- Business data models
- Business events
- Database domains
- DBMS data models
- Missions
- Organizations
- Schema data models

The names for these project template examples were all taken from the Knowledge Worker Framework and other Whitemarsh materials. The goal here is to classify the possible types and kinds of projects that are likely to be accomplished within the organization. This classification does not, however, relate to an enterprise's resource, that is, Finance, Human Resource Development, Distribution, Manufacturing, etc. Rather, these project templates serve as the mechanism to classify projects that address, for example the development of a Business Data Model within the enterprise resource, Human Resources.

Associated with each project template there are Deliverable Templates and Task templates. These are independent of Project Templates so that standard Deliverable Templates and Task Templates can be developed and employed within Project Templates. For example, every Project Template that creates a Knowledge Worker product is likely to contain a presentation. For any presentation, the deliverable is likely to be the same: that is, the presentation's materials. Additionally, the work plan to create the presentation is also likely to be the same. To create the Project Template, presentation, the Deliverable Template and Task Template merely have to be associated to the Project Template by creating two intersection entity instances, Project Template & Deliverable Template, and Project Template & Task Template. This same multi-use strategy is likely to apply to project reviews, testing scenarios, user guide development, project plans, phase plans, and the like.

The deliverable template entity is recursive and thus can contain subordinate deliverable templates. For example, a deliverable template may be the business data model. Contained within it would be the database domains, database objects, data elements, and business entities. Within business entities would be the entity, attributes, primary keys, and foreign keys. In this regard, the set of meta-entities from the Whitemarsh metabase are clearly candidates for deliverable templates.

In addition to deliverable templates, there are also deliverable template types that represent the classification of deliverable templates. For example, there might be a deliverable template type such as:

- Process models

- Data architectures
- Business information system architectures
- Plans
- Analyses

Each of the deliverable template types is a way to broadly classify the different types of deliverable templates to which each belongs.

Task templates are recursive and represent the natural hierarchical decomposition of tasks. For example, there might be a task template for defining database domains. The subtasks within that task template might include identifying, describing, reviewing, and finalizing the database domains. The complete set of tasks and subtasks for the Whitemarsh database project methodology is provided in a Whitemarsh work breakdown structure book[8].

Collectively, the project, deliverable, and task template entities enable the standard definition of the types of projects that are performed within the enterprise. By standardizing the project types, management and status reports can be created.

Deliverable templates contain an attribute for unit effort hours. This value represents the quantity of staff hours required to complete the deliverable under normal conditions. A deliverable, for example might be a defined data element, and the deliverable unit effort for this might be 2 staff hours.

The deliverable template entity also contains another attribute that indicates whether the work associated with producing the deliverable is divisible. For example, it is not likely that the work to create a defined data element is divisible. Therefore, no more than one person should be assigned the task of creating a single defined data element. Similarly, task templates contain an attribute that indicates whether the task is divisible. The task (but not the task template) contains an attribute to indicates the quantity of deliverables to be produced. If the task template indicates that the task is divisible, and if the quantity of delivered units is more than one, then the work is divided as evenly as possible among all assigned staff. The rules for work division are these:

Deliverable [Template] Divisible	**Task [Template] Divisible**	**Task deliverable quantity**	**Assigned work divided?**
No	No	N/A	No
No	Yes	more than one	Yes
Yes	Yes	one	No

[8] Great care must be exercised as to the level of detail for tasks. If they are too shallow, ambiguity results. If they are too deep, resentment and strangulation occurs. In the Whitemarsh database project methodology, for example, the table of contents of the work breakdown structure represents the right level of detail by which a Whitemarsh database project should be specified and managed. The Table of Contents is 14 pages. The full work breakdown structure is 125 pages.

Deliverable [Template] Divisible	Task [Template] Divisible	Task deliverable quantity	Assigned work divided?
Yes	Yes	more than one	Yes
Yes	No	N/A	No

Whenever work is not divisible, the work's elapsed time is computed to be that of the slowest working person. Other assigned persons are presumed to be "watching" the expert teach or lead the others in the production of the deliverable. Hence, the work proceeds at the rate of the slowest person, not the fastest.

3.4 Staff

Figure 8 has the following entities grayed:

- Job Title
- Phone
- Phone Type
- Skill
- Skill Level
- Skill Level Type
- Staff
- Staff Skill Levels
- State

The staff entity contains the necessary information to identify persons associated with projects. Staff can be associated with more than one project. The staff member's name, address and phone numbers are entered. In addition to this basic information, hourly costs are stored. These exist as direct hourly costs and indirect hourly costs. Direct hourly costs are typically those associated with salary and benefits. Indirect hourly costs are those associated with assigned overhead such as office space, tools, management and supervision, and the like. When staff costs are computed for a project, both types of hourly costs are employed to produce subtotals and then overall staff cost.

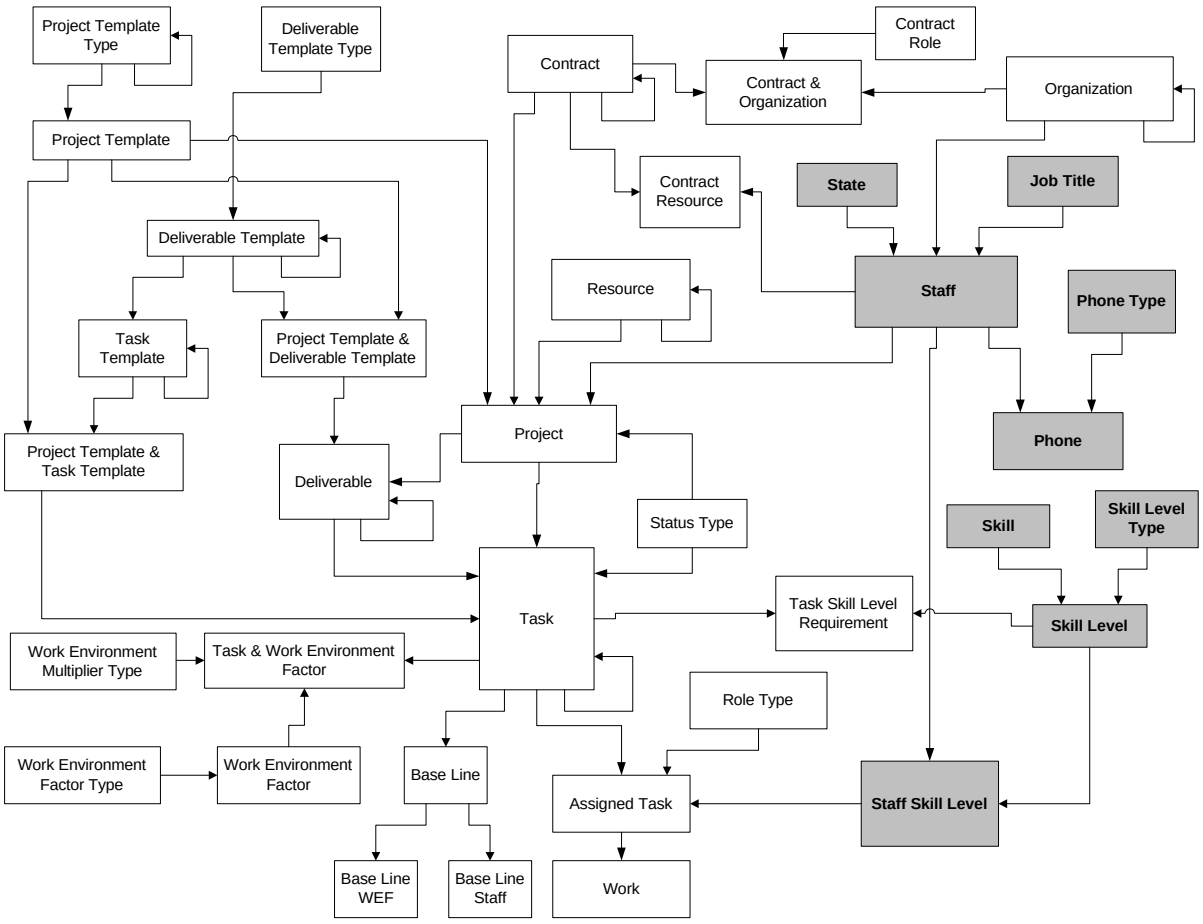

Figure 8. Staff

The staff member's skill levels are identified from a list of skills and the skill level types. The skills should relate to the type of activities performed. Skills might include for example:

- Data Administration
- Data Quality Control
- Database Administration
- Documentation
- Logical Database Design
- Management
- Physical Database Design
- Programming
- Quality assurance and quality control
- Requirements analysis

- Systems Analysis
- Teaching
- Testing
- Training program development

With respect to each skill, there can be distinct types that characterize the skill level achieved for the skill. The following skill level types are suggested:

- Expert
- Journeyman
- Novice
- Unskilled

A skill level is a skill of a certain performance type. For example, unskilled data modeler, novice data modeler, journeyman data modeler, or expert data modeler. For each combination of skill and skill level type, a multiplier is affixed. This multiplier field represents an assessment of the relative speed that work can be performed. For example, if the norm is journeyman, the Skill Level Multiplier might be classified as 1.0. If a novice performs the task 30% slower, the Skill Level Multiplier might be 1.3. For an expert who performs the work 20% faster, the Skill Level Multiplier might be 0.8. Finally, an unskilled worker might take twice as long and would have a Skill Level Multiplier of 2.0.

Once a complete set of skills, skill level types, and skill levels with their multipliers are created, they can be associated with staff. A staff member may have more than one skill, and be an expert at one and a novice at another. Once a person through their staff skill level is assigned to work on a project task, the staff skill level multiplier either speeds up the work, has no effect, or slows it down. This assignment capability provides project managers the ability to explain exactly why two projects that are the same in all respects except for the skill levels of the assigned staff will take two different quantities of staff hours.

3.5 Project Building and Estimation

Figure 9 has the following grayed entities:

- Assigned Task
- Base Line
- Deliverable
- Project
- Role Type
- Status Type
- Task

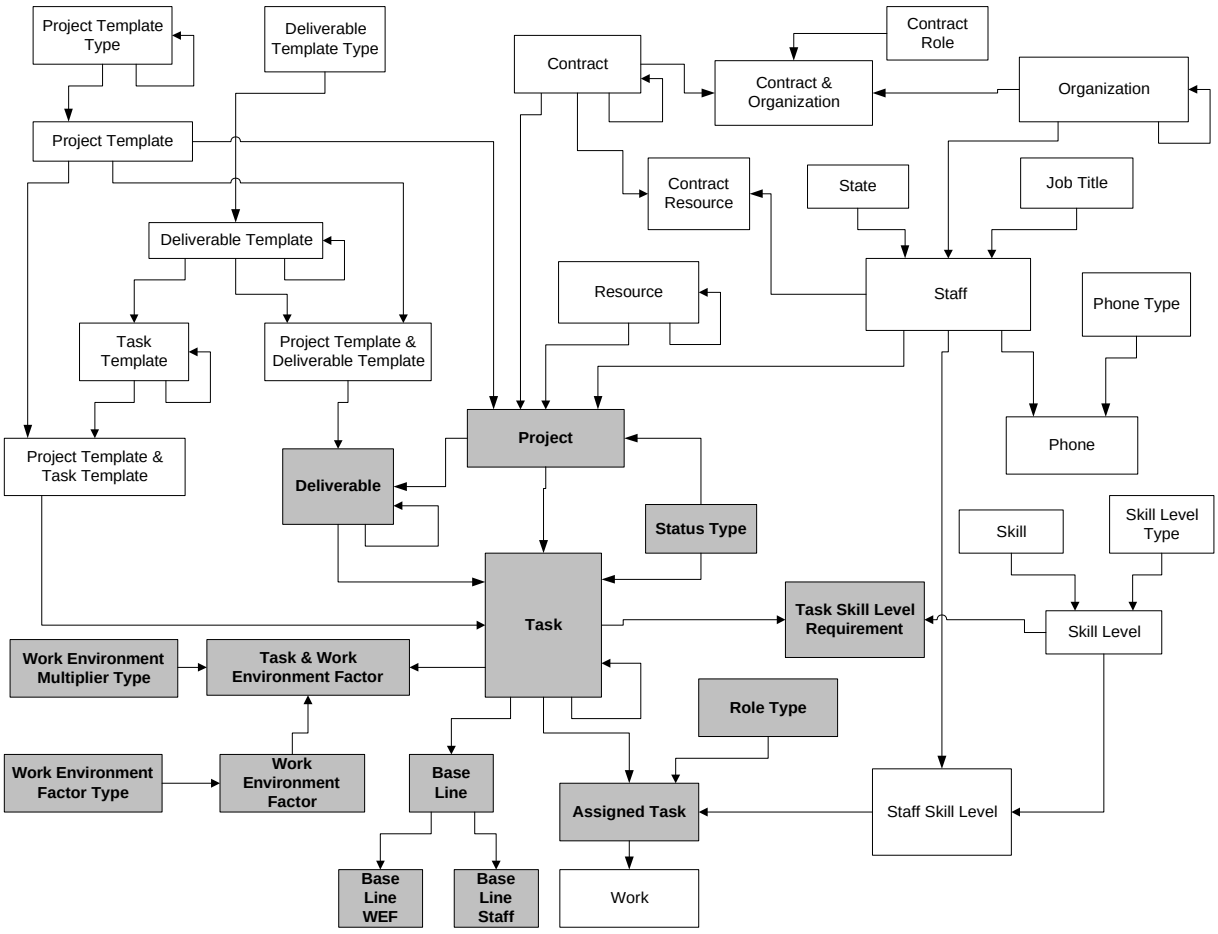

Figure 9. Project Building and Estimating

- Task & Work Environment Factor
- Task Skill Level Requirement
- Work Environment Factor
- Work Environment Factor Type
- Work Environment Multiplier Type

To create a project, the following steps must be accomplished:

1. Create the project record
2. Choose the project template
3. Choose the project's status
4. Choose the project's resource

5. Select the project's contract
6. Enter the project's description
7. Choose the project's leader
8. Create basic information for each task
9. Choose the appropriate work environment factor for each task
10. Assign the tasks to project staff members
11. Choose the role that each project staff member performs on a task
12. Press the Compute Project Resources button on the project record
13. Record the project's base line

In Step 1, the project record is created by an Insert button on the screen that contains the list of existing projects. Upon insert, a new [but null] project is created and is presented for further update.

Step 2 requires the selection of the project's project template. A list is presented and one is chosen. Step 3 consists of supplying the project's title. In step 4, a status is chosen from the status type entity list. These statuses should proceed through a life cycle. The following sequence of statuses are suggested:

1. Proposed
2. Deleted
3. Approved
4. Scheduled
5. On-going
6. Terminated
7. Completed
8. Reviewed
9. Finalized

In Step 5, the appropriate contract under which the task is performed is selected. In Step 6, a 250 character description of the project can be included. While this seems short, it must be remembered that there are many other name and description attributes that in combination serve to fully describe both the project and its context. Included for example are all the deliverables and their templates, tasks and their templates, contract, and resource.

In Step 7, the project's leader is chosen from a staff list. At this point, the only additional piece of information that can be added at the project level is the project's start date. Once this is entered, all remaining work must be done at the individual task level. This is done in steps 8, 9 and 10.

Once the project's basic information is captured, the project update window is closed. The list of projects is then re-presented. Highlight the newly created project and then press the Build Project button. Immediately, all the deliverable template records and task template records

associated with the identified project template record are copied into the project's deliverable and task entities. At that point, the lists of deliverables and tasks can be displayed

Once the project's build step is complete, the project's tasks can then be updated. This process is started by highlighting the project from within the project list and pressing the Change button. The project's update window is then presented. Each project task is changed by first highlighting a specific task in the project's task list window and then pressing the Change button.

> ***As a consequence of the Whitemarsh strategy of project management, no task may be deleted from, nor can any additional tasks be added to any project. If there is additional work that is in the nature of additional tasks then the project's template must be changed. That is, new tasks added to or existing tasks deleted from that task template from within the project's template.***

The ability to add or delete tasks from a project was specifically ***removed*** from Whitemarsh project management because to allow additions or deletions would in effect be defeating the Whitemarsh project management strategy: standardization of project specification, accomplishment, and measurement. If task lists derived from a template are allowed to be modified then eventually all templates will be irrelevant as all projects will ultimately be uniquely accomplished.

Great care must therefore be taken to ensure that the task lists at the project template level are sufficiently comprehensive but not too detailed. The exact details of how a task is accomplished should be left to the ingenuity of the person performing the work. That is what distinguishes the skill types, inexperienced, novice, journeyman, and expert. Having tasks at such a level of detail that every five-minutes of effort is both specified and controlled is both suffocating and absurd. Whitemarsh project management is for knowledge workers, not for process or manufacturing-line workers.

As an example, if the project is an IT project that creates a database application, and if the Whitemarsh methodology, which has a 125 page 3300 task work breakdown structure, is employed, these 3300 tasks must not be stored in the project management database. But, if by mistake they were, then it would take forever to estimate such a project and it would also be impossible to manage because the task detail is too low. Instead, only the high level tasks should be stored. This represents about 450 tasks. That's a 8:1 reduction.

Better still, however, would be to divide the database project into at least six smaller projects, one for each phase. In that case, the six projects would represent only about 70 tasks each. That's much easier to estimate and to manage.

The only tasks that can be changed are those that are "leafs." A request to change a non-leaf task results in an error message. In this Step 8, the basic task information can be adjusted. At the outset, the task window automatically shows the related task template, deliverable template, and the deliverable template's unit effort hours. This data is automatically brought from the task template. Automatically loaded as well is the task sequence number, whether the task's work is divisible, and the task's title. The user is able to change the task's title and description to

something more stylistic, detailed, or appropriate. Notwithstanding, the task should still be seen as an implementation of a task from the task template.

Entered on the task window is the current status (1=Proposed through 9=Finalized), and the quantity of deliverable units that the task expects to produce. For example, 125 defined data elements. Once the quantity is entered, tabbing off the field causes the task level unit effort hours to be automatically computed. If a quantity of zero is recorded, then the task will effectively not participate in the determination of the project's resources. The task's start date should also be entered and as it is used to determine the task's completion date after work environment factors and staff are assigned. Note however, that the task's start date is "overruled" when the schedule for the entire project is computed.

In step 9, the appropriate work environment factors that govern the tasks are picked. On the task screen there is a list box that shows chosen work environment factors. When the Insert button for that list is pressed, a set of the work environment factors is presented. Choices should be made that relate to the work environment for that specific task. Each chosen work environment factor has an its effect on the unit effort hours computed for the task.

The staff hours in the task templates assumes normal work environments for each task. While the definition of the work environment factor types and the specific factors is clearly under the control of the project management administrator, the following suggestions are offered:

Factor Name	Factor Value	Factor Description
Reviews conducted by the client	1.00	No effect
	1.00	Reviews, that is, walk thru's are conducted by the client as scheduled
	1.10	Reviews are conducted but by inexperienced reviewers
	1.20	Reviews are generally not conducted
Equipment available for analyst/programmer	1.00	No effect
	1.00	Workstation connected with shared case/metabase environment
	1.10	PC with stand-alone case tool environment.
	1.25	PC with no case/metabase environment

Factor Name	Factor Value	Factor Description
	1.30	For no equipment available to the staff, except through an administrative person
Equipment outages	1.00	No effect
	1.00	If the equipment and all required software is available for at least 6 business hours each day
	1.16	For each hour below the average of six that the equipment is unavailable
Extent of user contact	1.00	No effect
	1.00	If the users are available within half day request to review interim products (work in progress, but not a deliverable).
	1.10	If users are available but only within 2 business days of request
	1.20	If users are available but only within 4 business days of request
	1.30	If users are available but only within 6 business days of request
	1.40	If users are generally not available

In Step 10, staff members are assigned to the task. The list of currently assigned task members appears on the task screen. Changes in these assignments is accomplished through the Insert, Change or Delete buttons below this list.

When the Insert button is pressed, a list of staff members and their associated skills is presented. When a specific staff member is chosen, so also chosen is the staff's skill and skill level multiplier.

Once a staff member has been assigned to a task, Step 11 consists of picking their appropriate role. Suggested roles might be:

- Developer or contributor
- Manager or leader
- Reviewer

Once all the work environment factors and staff have been assigned to a task, the resources associated with that task can be computed. This is done through three buttons:

- Compute Factored Hours
- Compute Elapsed Hours
- Compute Completion Date

When the Compute Factored Hours button is pressed, the standard hours from the task's task template is multiplied by the multiplicative summation of all the work environment factors producing Factored Hours.

When the Compute Elapsed Hours button is pressed, the staff work hours for each assigned staff member is computed as the Factored Hours multiplied by the staff's Skill Level Multiplier. The elapsed hours for a task is determined as the longest effort for any one participant in a task. The field, staff assigned hours, is the commutative quantity of all staff work hours.

When the Compute Completion Date button is pressed, the start date of the task is employed to then compute the completion date according to these rules:

- Duration Days are computed by dividing elapsed hours by hours-per-workday. The Whitemarsh project management software's basic reference data can be changed. It includes, for example, hours-per-workday. If a task has a duration of 24 staff hours and if the hours-per-workday are 6, then the staff-days is equal to 4.

- The total staff-hours for a project is the sum of all the staff hours.

- The completion date depends on whether a task which contains other tasks is marked as serial or parallel.

  If the containing task is marked as serial, then the completion date for its contained tasks is computed to be start date for the first contained task plus the duration days for every other contained task. Weekends and holidays are excluded.

  If the containing task is marked as parallel, then the completion date for its contained tasks is computed to be the start date for the first contained task plus the duration days for the longest contained task. Weekends and holidays.

If more precise scheduling is required, the Whitemarsh task and resource data can be exported to an ASCII file and then imported into a project management system like Microsoft Project. Once the appropriate PERT chart is entered into MS/Project and the tasks are serialized or are parallel are identified, then the project's schedule can be computed. A Whitemarsh Project module

allows for the re-importation of the start and end dates for tasks and an indicator is set in each task to show that the start and end dates have been set externally.

In step 12, the overall project resources and end-dates are computed. Once all the work to each task has been completed, the Compute Project Resources button is pressed. At that point, the project's factored hours, staff hours, and elapsed hours are all computed. The start-date and end-dates for each task are recomputed beginning with the project's start date.

Unless otherwise indicated, tasks within a project are processed in a hierarchically serial manner. If a project contains multiple tasks, or if a task contains other tasks, and the schedule-parallel indicator is parallel, then the contained tasks within the project or within specific tasks are scheduled in parallel. That is, the elapsed time for a containing project or task is the elapsed time of the longest contained tasks. Once all the scheduling is accomplished, the project's end date is computed. If the OK or Close button is then pressed the data is stored in the project management database.

Once a project is scheduled, the final button, Check Overload, can be pressed. This button triggers an analysis that determines if any staff member is overloaded for a task. It determines this by summing the staff members assigned hours across task schedules. If the sum is greater than the allowed work hours for the period, then the overload indicator is set. A staffing allocation report is then generated and it shows the project and task level of staff loading by staff person across time. The granularity of time used to assess overloading is one week.

Step 13 enables the project manager to record a baseline for the project. A baseline is the dated permanent recording of key project, task, resource and schedule information. When the baseline is recorded, a new set of the information is created for every task in the project. Collected as well is the key information from the work environment factors and the staff assignments on the date of the baseline.

Once project planning is finished, the project's baseline is created by pressing the Baseline button on the project update window. The data values for all of the above attributes are collected on a task by task basis and stored into the baseline entity. Baselines support progress reporting. Since a key attribute of the baseline entity is the baseline date, multiple baselines can be created and the baseline information can be described and tracked during the life of the project.

3.6 Project work

Figure 10 has only one entity grayed, Work. This entity records the hours expended by an assigned staff person including a description of the work performed. Work is recorded in terms of the assigned task. A window that contains four interlinked lists is presented. The first list contains the projects. The second, linked with projects is the contained tasks. The third is the task's contained assignments, and finally the fourth contains the work records. If a new work record is to be added, the Insert button is pressed. The recorded work includes the start and end dates, hours worked, quantity of deliverable units completed, and the description of the effort.

As a task is completed, its status should be changed. As the project is finished, the project's status should also be changed. Reports perform project accomplishment analysis by schedule, status, hours, and earned value. Earned value analysis computes the percent of a task being complete based on the deliverable units accomplished. If 100 units are proposed and 50 are complete, then percent complete is judged to be 50%. Earned value date adjustments mean that the quantity of hours expended by summing work hours to complete a task results in projections of the hours needed to complete the task. When earned value rescheduling is performed the task's end-date is recomputed. On the project level, earned value analysis and rescheduling accomplishes the rescheduling of the entire project. This results in new task and project dates.

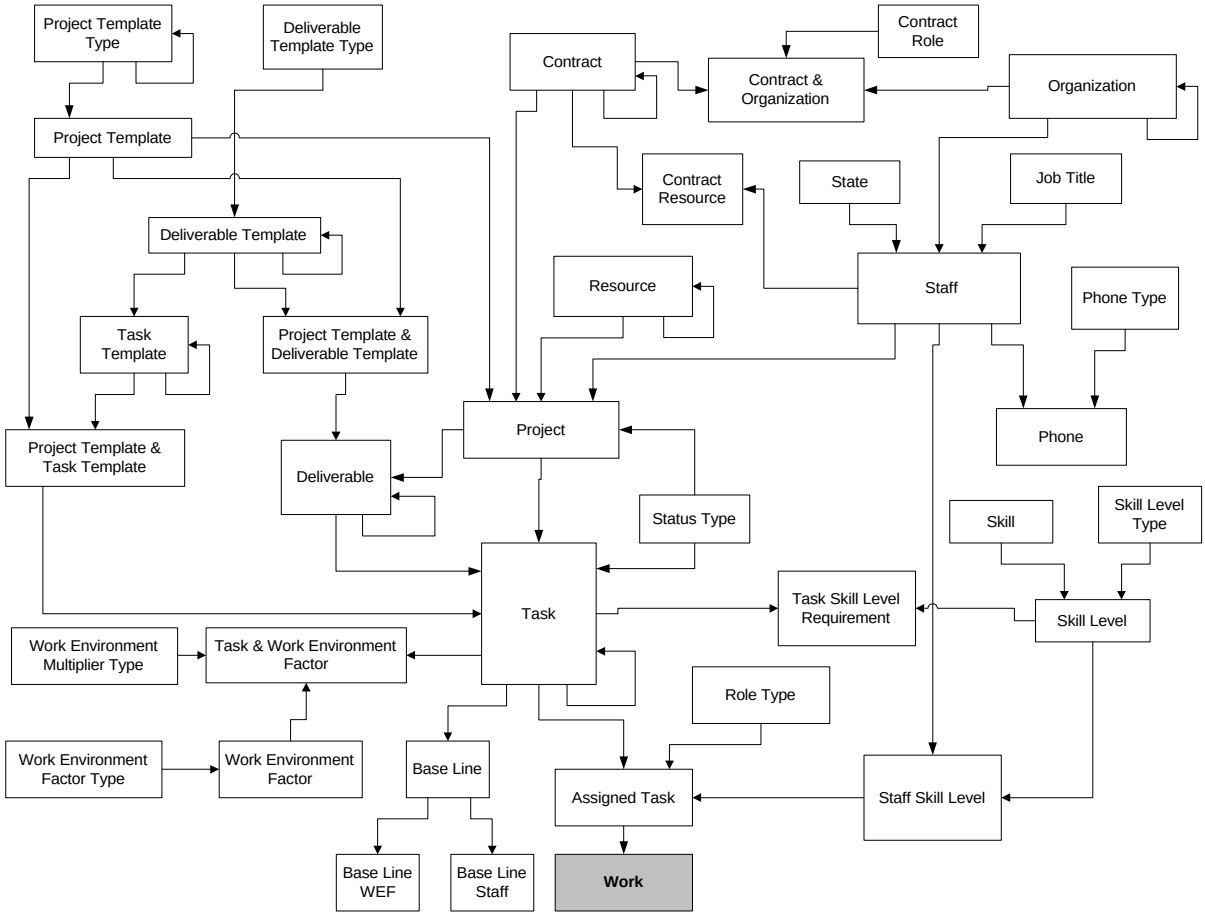

Figure 10. Project Work

4.0 Self Learning

A Whitemarsh project management module performs an analysis of the work-hours data contained in the Work entity and compares it to the originally supposed work-hours inferred for a deliverable-template from within the associated task-template instances. A subprocess then enables the reviewer of this analysis to selectively update the originally conceived staff hours durations.

Index