AN OBJECT-ORIENTED DATABASE FOR PROJECT COST COMPILATION

Chul Soo Kim, Ph.D.
Assistant Professor
Department of Construction Technology
Purdue School of Engineering and Technology
Indiana University-Purdue University Indianapolis
Indianapolis, Indiana 46202, USA
cskim@iupui.edu

ABSTRACT

Larger construction companies generally have several on-going projects at one time, many internationally. The ability to make corporate decisions in a timely manner for these companies becomes critical to avoid costly corrective measures. The information necessary for making such decisions comes from a large pool of corporate data. In order to efficiently process large amount of data, an information model for multiple project management is essential.

In this research, an information model based on *Cost* is designed and presented (OCIM, Object-oriented Cost Information Model). A prototype system is developed based on the model using an object-oriented database. Using the periodical update feature of the prototype system, an immediate access to necessary information by the user is made possible. The different levels of information abstraction are also achieved to meet the different level of the user.

KEYWORDS: Construction, Cost, IT, Database, Object-oriented, Information, Multiple Project Management

1 INTRODUCTION

A construction company has to carry out and manage several projects at a time. Larger firms deal with hundreds of projects spread nationwide and often worldwide. As the number of projects within a firm increases, it becomes difficult for top management to keep track of progress, and thus, to control the projects. The difficulties come from the complexity of collecting, processing, and utilizing information for managing multiple projects.

The construction industry benefits from the advancement in modern information technology (IT) in computerizing many practices otherwise performed manually. Many software applications have been actively adapted for construction management; however, most of them only support a single project. A system that can integrate multiple project management information has become necessary for many construction firms where corporate decision makings over several projects are critical.

In order to provide information necessary for making corporate decisions, a well integrated information system is essential. The system must be able to deal with the complexity of the construction information as well as the large number of individual organizations participating in the projects. Further, the information should be available whenever and wherever because failure to make timely decisions means potential financial losses.

In this research *cost* is used as a parametric measure of progress and other project performances [2]. Various forms of project costs are modeled for the database development. This research proposes *Mapping Objects* which periodically process project data into multi-project information. A *Mapping Object* is a neutral connector between separate modules that collects compiled data from corporate database.

An object-oriented database is used to properly demonstrate the proposed information model [3]. A cost information model for the database is designed with maximum efficiency in large data compilation as its primary objective.

2 COST INFORMATION MODEL

Cost is used as an important element of the information model in this research. *Model* refers to the complete view of all the elements in an information system. An information model of construction cost is developed for the database system.

2.1 Cost Information

Generally speaking, *cost* is a monetary value of an item. It is used as an indicator of business performance. Daily progress, project performances, and regional business status can be all represented in dollar amounts [2]. This research organizes cost information along the three axes of (1) Cost Type, (2) Cost Lifecycle, and (3) Change Order as illustrated in Figure 1.

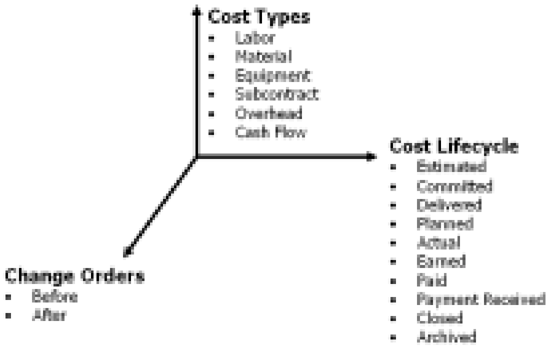

Figure 1. Three Dimensions of Cost Information

Cost Type of a cost item can be one of the six cost types: labor, material, equipment, subcontract, overhead, and cash flow (or payment). On the other hand, the cost item goes through its own *Cost Lifecycle* of estimated, committed, delivered, planned, actual, earned, paid, payment received, closed, and archived stages. The same cost item also can be changed during the construction through *Change Orders*.

2.2 Cost Information Model

A model is a representation of structure. Within the context of this research, a model can be described as a complete mathematical and descriptive representation of an information object and its relationship to other objects. A project information model is a model expressed in terms of the project variables of cost, time, resources, and scope of work [4].
The modeling is used to assist in understanding and communication the requirements for the development of the information system.

2.3 Object-oriented Cost Information Model (OCIM)

The OCIM consists of four modules (layers); Cost Object Layer, Project Layer, Information Output Layer, and External Interface Layer. Figure 2 shows the overall layout of OCIM.
The Cost Object Layer (COL) contains basic cost information consisting of a *Cost Object* (CCostObjectCell), six sub-classes of the cost objects (CLO, CMO, CEO, CSO, COO, and CPO) representing *Cost Type*, and a *Mapping Object* (CCostObjectRoot).

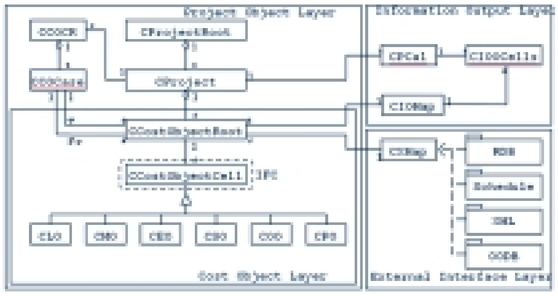

Figure 2. The Overall Layout of OCIM

The Project Object Layer (POL) includes the Cost Object Layer (COL) through the aggregation association between CProject class and CCostObjectRoot class, represented by a line with a diamond symbol at the one end. This is an important relationship because most of the information abstraction occurs here. External programming routines that selectively aggregate cost information according to the user's requirements are implemented in this relationship.
The Information Output Layer (IOL) contains a platform specific graphic output module which simply renders data in chart formats. The classes within this layer associate with the mapping object (CIOMap, or CInformationOutputMap), which relates the Information Output Cell Object class (CIOOCell) to Cost Object class.
The External Interface Layer (EIL) is the gateway of the OCIM to the external systems. The external systems are any database systems, software applications, or information standards over which the OCIM has no direct control. Yet the input from the systems is essential for the model. Some examples of these systems include scheduling systems, estimating software, aecXML, and other corporate financial database systems. A mapping object takes controls the information traffic between external systems and the OCIM, directing it to proper places. Separate data interpreters are necessary to import and export data between the systems.

3 OBJECT-ORIENTED DATABASE FOR COST COMPILATION

3.1 Cost Object Layer

Cost Object (CO) is a general term for the combination of objects in which most of cost related data is defined. It consists of three, hierarchically organized, primary objects: Cost Object Root (COR, i.e. CCostObjectRoot), Cost Object Cell (COC, i.e. CCostObjectCell), and the sub-classes of COC representing six cost types: labor, material, equipment, subcontract, overhead, and payment (i.e. LMESOP).

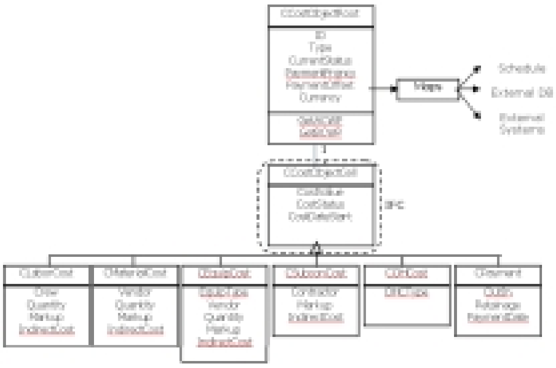

Figure 3. The Objects in Cost Object Layer

The objects in the Cost Object Layer (COL), COR, COC, and LMESOP, are the core objects of the proposed information model and they are represented in Figure 3. The COL is a complete package of cost data, a module with essential attributes and basic relationship definitions. Also, the data contained within the COL are, in essence, very plain values of cost and date with a very versatile nature. Thus, the COL has great potential as a module that can be plugged into other cost oriented information systems and utilized as a core element of many database systems in the future.

3.2 Project Object Layer

The Project Object Layer (POL) processes the data stored in Cost Object Layer and organizes the information at different abstraction levels. The aggregation association between the CProject and CCostObjectRoot classes is where most of the cost data processing occurs.

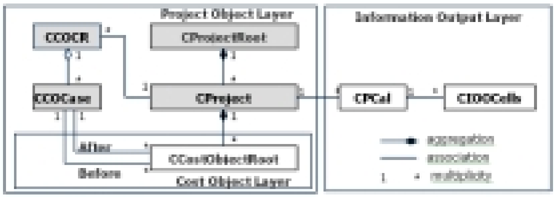

Figure 4. Project Object Layer

As shown in Figure 4, POL contains the objects for change order management. Also, CProject class has an association to CProjectCalendar class to provide the Information Output Layer (IOL) with access to project data.

The Project Object (represented as CProject) is equivalent to a single project and is a class in which the general project data is stored. It has attributes for construction type, project region, project period, and object identification. Some of the cost related data such as project tax rate, markup rate, and project fixed overhead costs also can be stored in this object.

CProject manages change orders through an association with Change Order Case Root (i.e. CChangeOrderCase) object. It also has a direct association to the Information Output Layer.

The Project Root Object (represented as "CProjectRoot") aggregates individual project objects (i.e. CProject) and organizes them into project information by project region and construction type.

Change orders are unavoidable in almost all construction projects. The revenue increase resulting from change orders often contributes to a considerable portion of the entire project revenue. Project managers must expect frequent change orders during any project and be prepared to accommodate the changes into their system from the beginning.

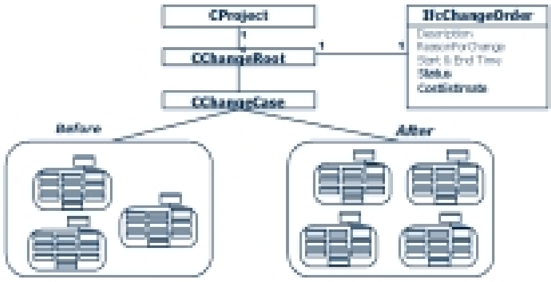

Figure 5. Change Order Object and Its Relationships

As shown in Figure 5, each change order becomes an instance of a Change Order Object (CChangeRoot) which consists of Change Order Case Objects (CChangeCase). A Change Order Case Object is a simple class with two associations, each pointing to a collection of cost objects. CChangeOrderRoot has an association which points to the IfcChangeOrder class. General properties of change orders such as descriptions, note on change reasons, date and time, status, cost estimate, etc. are modeled in the IfcChangeOrder class. IfcChangeOrder class is an externally defined class by the International Alliance of Interoperability [5].

3.3 Information Output Layer

The function of the Information Output Layer (IOL) is to transfer processed information to the user in a graphical format. It stores numeric values in a cell (i.e. computer memory cell) which represents a calendar date. The value stored is then plotted as a value on the vertical axis, with many values forming to form a connected line of graph. There are two objects defined for this operation: Information Output Cell Object and Project Calendar Object. Figure 6 depicts how data from CProject is compiled into Project Calendar Object.

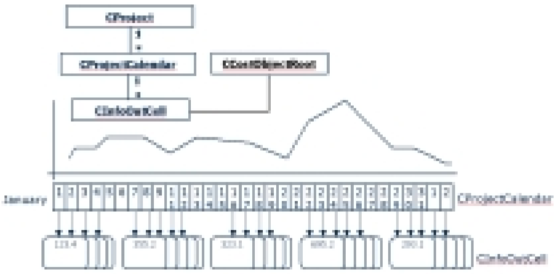

Figure 6. Information Output Layer

In the database, Information Output Cell Object is defined as a class named CInfoOutCell. The CInfoOutCell contains three simple attributes: value, cost type, and information type. Each CInfoOutCell class has two relationship pointers, one to Project Calendar Object (CProjectCalendar) and the other to Cost Object Root (CCostObjectRoot).

Project Calendar Object (CProjectCalendar) is a simple set of objects which represent calendar dates of project duration. Each instance of CProjectCalendar has associations to a number of numeric values (i.e. instances of CInfoOutCell) so that a certain value of a given date can be retrieved directly.

3.4 External Interface Layer

External Interface Layer (EIL) is a group of objects that provides an interface with external systems. It consists of a mapping object (CExternalMap) and an Activity object. The external systems considered for the development of the OCIM are project scheduling systems, corporate managerial database system, the aecXML standards, and IFC standards. An interface with the object-oriented activity object developed by Chin [1] is also built into the layer. Figure 7 depicts the interfaces between CExternalMap and the external systems.

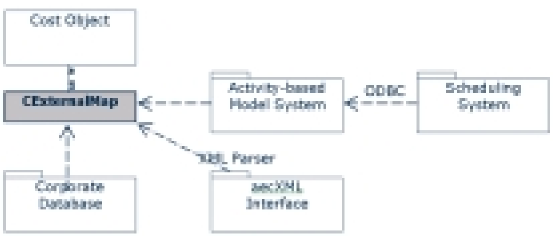

Figure 7. Objects and External Systems in External Interface Layer

3.5 OCIM Information Model

The OCIM is developed upon the criteria specified by the requirements of the management levels. It is important that the designed model is capable of producing the prescribed information. Most of the processes are external of the model itself and implemented as program routines in the prototype system (OCIMPS). The fundamental information retrieval process for this is also described in this section, along with how the information is organized to meet the different requirements of the four management levels. The four management levels are defined as: Operational Manager, Project Manager, Regional Manager, and Top Manager. While an operational manager and a project manager need detailed cost information of a single project (Figure 8), a regional manager and a top manager of the company would want compiled information from multiple projects (Figure 9).

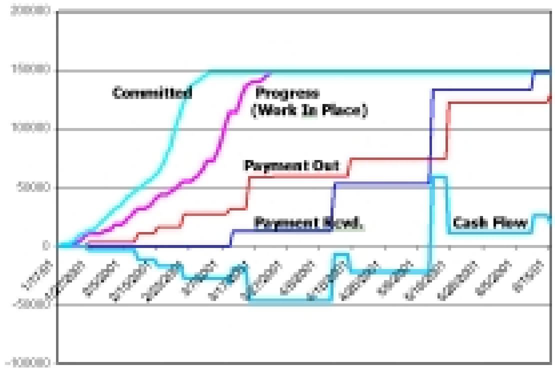

Figure 8. Cash Flow of Single Project

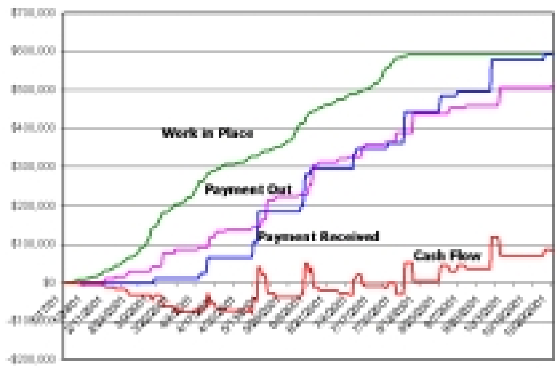

Figure 9. Cash Flow of Multiple Project

The information needed by top managers can be organized by aggregating the instances of CCostObjectCell. For the case of budget trends and forecasting, the instances of *Budgeted (or Planned)* cost are aggregated and organized per month or quarter year.

Figure 10 depicts the process of information retrieval. For Quarter 1 of the year 2004, there are three cost incurring interactions (i.e. instances of CCostObjectCell) of which cost values are at $12,000, $8,000, and $15,200. The total budget of $35,200 for the quarter is organized by the Project Object (CProject) and then passed into Project Calendar Object (CProjectCalendar). The value of $35,200 with the time indexed at Q1/2004 (i.e. the Quarter 1 of the year 2004) is stored within the output cell (CInfoOutCell) in the Information Output Layer (IOL). By repeating this process, a series of display data sets are prepared in a very compact format. When there is a query from a user requesting “Budget Trend for the Year 2004,” the prepared data set is used in the information output.

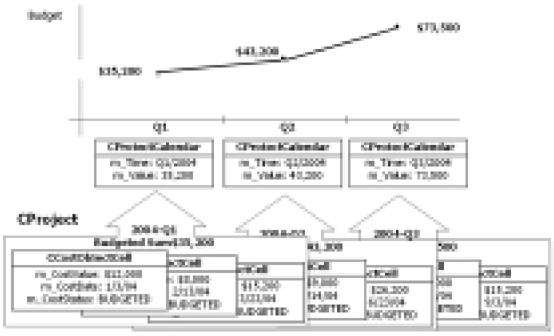

Figure 10. An Example of Information Retrieval Process

3.6 OCIM Prototype System

Based on the information model, a prototype system is developed. It is developed using ObjectStore™ PSE Pro as the object-oriented database engine and C++ language as a tool for developing user interfaces.

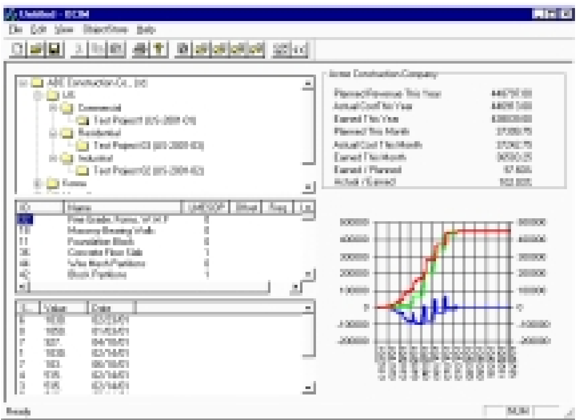

Figure 11. OCIM Prototype System’s Main User Interface

While the database management system, ObjectStore PSE, takes care of data maintenance, another module within the prototype system processes the data and provides the output to users. This module refreshes and updates the output data periodically so that the display of the information can be instantaneous. The user may choose a portion or an entire project, a group of projects, or the entire projects of the company depending on his or her user level. Especially for the top management, the amount of data to be processed for a single display will be enormously large and a real-time processing often can be difficult. The prototype system always contains an updated set of data for immediate response to the user’s needs. The display data, therefore, will have a certain amount of lag (i.e. 1 day) in terms of how current the data is. However, such lag is widely accepted in construction without too much concern in making managerial decisions. Yet the overhead for maintaining such data is at minimum because the display data consist of simplified numbers and dates.

3.7 Information Abstraction

As described in Section 3.5, four levels of information abstraction have been specified. At the operational level of management, the cost information must be highly detailed and accurate. It must contain every cost related interaction of material, labor, and equipment related expenditures. The collection of such expenditures then can represent the progress of construction activities. The progress of the work packages and ultimately of the entire project can again be derived from the collection of detailed single cost interactions.

Figure 12 through Figure 14 show how the information is organized for different managerial levels. The cash flow profile of one project is prepared for a project manager as shown in Figure 12, while the charts of multiple projects are generated for regional managers in Figure 13 and Figure 14. The corporate level cash flow trend and forecasting can be further generated using similar methods.

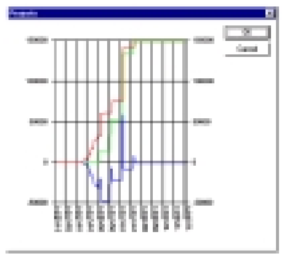

Figure 12. Cash Flow of One Project

Figure 13. Cash Flows of Regional Projects

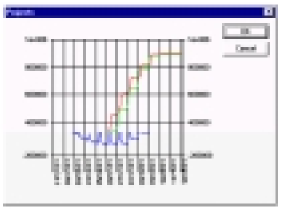

Figure 14. Cash Flow of All Projects

4 SUMMARY

The ability of larger construction companies to make corporate decisions in a timely fashion becomes critical to avoid costly corrective actions. The information necessary for making such decisions comes from a large pool of corporate data. In order to efficiently process large amount of data, an information model for multiple project management is essential.

In this research, an information model based on *Cost* is designed and presented. An operable prototype system is developed based on the model using an object-oriented database.

Using the periodical update feature of the prototype system, an immediate access to necessary information by the user is made possible. The different levels of information abstraction are also successfully generated to meet the different requirement of the user of different levels.

REFERENCES

[1] Chin, S. An Integrated Lifecycle Information Framework for Construction Project Management. Ph.D. Thesis, Department of Civil Engineering, University of Illinois at Urbana-Champaign, 1997.

[2] Fleming, Q., and Hoppelman, J., *Earned Value Project Management*, Project Management Institute, Newton Square, PA., 1996.

[3] “The User Guide-ObjectStore PSE Pro for C++, Version 3.0,” HTML Format Document, ObjectDesign, Burlington, MA, 2000.

[4] Westney, R. E. *Computerized Management of Multiple Small Projects*, Marcel Dekker, Inc., New York, N. Y., 1992.

[5] “IFC End User Guide-Industry Foundation Class Release 1.5,” International Alliance for Interoperability.