會計評論
第26期，第140-158頁
中華民國八十一年四月

物件導向系統分析法在作業成本系統上之應用

吳琮璠
國立台灣大學副教授

摘要

本研究之主要目的在結合資訊系統與管理會計理論。本文首先分析物件導向理論運用於系統分析之特色與此軟體工程理論之優點，而後說明如何運用物件導向系統分析法以設計一個自動化的整合性作業成本系統。

作業成本會計制度乃一套強調自生產及提供產品所需之作業活動成本動因的角度，以計算產品成本之管理會計方法。在作業成本制度下，成本分析的基礎除了傳統與產量有關的成本動因，如直接人工、直接原料與機器小時外，尚包括其他與產量無關之成本動因，例如訂貨次數、運貨次數等。

作業成本系統的資料與系統功能包括複雜的物件或個體。本研究分析作業成本系統電腦化的結構與功能，並說明物件導向分析法應用於設計作業成本系統之過程與其優點。

THE CHINESE ACCOUNTING REVIEW
Vol.26 PP.141-158
APRIL 1992

Object-Oriented Systems Analysis Methodology – The Modeling of Activity-Based Costing Systems

Rebecca Chung-Fern Wu *
National Taiwan University

Abstract

ABC system is a system of product costing. It emphasizes costing activities that are performed to produce and deliver the products. The cost drivers, or allocation bases, are used as a measure of the quantity of activity consumed on the products. ABC systems use many allocation bases, including traditional volume- related drivers such as direct labor, material and machine hours, and non-volume-related drivers such as number of times setup, number of times ordered, and number of times shipped.

ABC data and operations performed on them and the need to consider complex objects as single entities have led to the view that the object-oriented approach might be better suited to the analysis and design of the ABC system.

This research attempts to integrate theories of managerial accounting and information systems. The paper explores the structure and behaviors of ABC costing data and what the object-oriented analysis methodology has to offer. Its application in the design of the automated ABC costing systems is demonstrated throughout the paper.

*Special thanks go to professor Chee Chow and Ann Wu, as well as two anonymous referees for helpful comments and suggestions.

1. Introduction

Many researchers have attempted to integrate theories of accounting and information systems, for example, Eaves (1966), Lieberman and Whinston (1971), Everest and Weber (1977), and many others. Everest and Weber (1977) pointed out that ".... There are a variety of factors that motivated these attempts. First, the practical impact of the computer on business data processing is significant enough that efficient ways of implementing accounting theories within the new technology must be studied...."

This paper examines some aspects of the recently developed systems analysis methodology: the object-oriented approach (Korson and McGregor, 1990). The use of the object-oriented analysis methodology in helping the design of an integrated and automated activity-based accounting system is demonstrated in the research. This study attempts to integrate this particular information system methodology and managerial accounting approach. A manufacturing implementation case of activity-based costing (ABC)system is used as an example to demonstrate such an integration.

2. Object-Oriented Methodology

The term **"object-oriented"** means that we organize computer software as a collection of objects (entities) that incorporate both data structure and operations on data. This is in contrast to traditional software design in which data and operations are only loosely connected.

Although there is still no accepted definition of object-oriented data model, it is generally recognized that the object-oriented paradigm has several major features that differentiate it from traditional data modeling methods. Five major concepts of the object-oriented data model are discussed in this section: **objects, generalization and inheritance, information hiding, abstract data type (ADT) encapsulation, and polymorphism.**

2.1 Objects

It is easier to grasp the nature of the object-oriented terminologies by using entity-relationship (E-R) model as an analogy. In the E-R model, the world is viewed as consisting of entities and relationships among entities.

Figure 1 depicts the E-R diagram. Rectangular-shaped nodes represent entity sets. Diamond-shaped nodes represent relationships. The relationship between SUPPLIER and PRODUCTS is Many (M) to Many (N), which means that a supplier may supply many products, and a product may be supplied by many suppliers.

Figure 1. An E-R diagram

Table 1 presents a summary of terminological analogies that can be drawn between the E-R data model and the object-oriented model.

The comparison represents more of a terminological analogy rather than a direct correspondence. An object corresponds to an entity; the attribute corresponds to the data item; the operation corresponds to the method, function or process; and the message passing corresponds to the relationship. The **message passing** makes use of established **relationships**. It allows a sending object to access the variables and methods of a related object, the receiving object. In the E-R data model, data are separated from processes, while in the object-oriented data model, processes and data are internal to objects.

In summary, an object can defined as follows:

Table 1
Terminological Analogy Between Object-Oriented Data Model and E-R data Model

Object-Oriented Data Model	E-R Data Model
Object	Entity
Attribute	Data item
Operation	Method, Function, Process
Message Passing	Relationship

OBJECT = DATA + METHODS + STATE BEHAVIOR

An object can be a physical entity or a concept. In the object-oriented paradigm, an object is described as more than an entity described in the E-R model. An object is described with **attributes** (or data), **methods** (operations on data, e.g. editing , updating, and reporting) and **state behavior**. For example, the **object** SUPPLIER has Supplier_ Id and Supplier_ Name attributes. **Object attributes** describe properties of an object. Attributes model the state of an object; **methods** define how an object changes its states, for example, updating data.

The **state behavior** in the object-oriented approach consists of the following object-oriented characteristics: abstract data type encapsulation, inheritance, information hiding, and polymorphism.

2.2 Abstract Data Type Encapsulation

Encapsulation means that the object description includes both its attributes and operational methods. For example, a SUPPLIER object description includes **data attributes** such as Supplier_ Id, Supplier_ Name, and Supplier_ Address, and **operational methods** such as Create, Edit, and Delete. An **abstract data type** capability means that there are methods that are specifically defined for an object.

2.3 Inheritance

When an object class Y inherits from class X. Class Y has, by inheritance, all the features of X. For example, the COST ACCOUNT object has the MATERIAL COST ACCOUNT, LABOR COST ACCOUNT, and OVERHEAD ACCOUNT objects as its subclasses. Property inheritance states that each subclass inherits both the general attributes and methods of the superclass, while adding special attributes of its own. Assume that MAINTENANCE LABOR is declared to be the child class of INDIRECT LABOR, then all attributes of INDIRECT LABOR class will be automatically contained in MAINTENANCE LABOR class. When defining the MAINTENANCE LABOR class, we only need to consider the attributes which have not been defined for the INDIRECT LABOR class. It makes programming codes of superclass reusable for the subclass and simplifies the coding for subclass.

2.4 Information Hiding

Information hiding means that operations with the object is written with two separable parts: the object interface and the object implementation. Users know only the interface. The separation of interface and implementation allows much of the maintenance activity to be hidden from users of the object. It also allows the object interface to be mapped to several different implementations.

Each object provides a uniform external interface to end users. Differences between objects are coded inside each object. This approach makes the programming codes easier to maintain.

2.5 Polymorphism

In the object-oriented paradigm, polymorphism refers to different representations of the same attribute based on its position in the generalization hierarchy. For example, the method of purchase discount calculation may depend on the subclass of BUYER, which may be either WHOLESALER or RETAILER.

The object-oriented approach differs from the traditional data processing method in several ways. Traditional data processing operations have

the feature of viewing a series of operations as a transaction unit, for example, an order entry transaction or an invoice payment transaction. It also has the feature of simple operations such as retrieval, insertion, deletion, and updating of data. In addition to having the above- mentioned features, the object-oriented approach has the state behavior features as described in section 2: an operation that traverses the generalization hierarchy, abstract data typing capability, etc.

In structured methodologies exemplified by Yourdon (1989) and DeMarco (1979), primary emphasis is placed on specifying and decomposing system functions. Such an approach may seem to be the most direct way of implementing a system. However, if the requirements change, a system based on decomposed functionality may require massive restructuring. In object-oriented analysis, the decomposition is based on the objects in the problem domain, so the developers of different programs in the same problem domain tend to discover the same objects. This increases the reusability of components from one project to another.

The implementation of the reusability of objects concept can be done by (1) establishing an object library that consists of object descriptions, (2) grouping objects by category, and (3) inheriting operation methods from other objects. Therefore, the new object needs to define only attributes or operation methods that are new or not yet defined.

The features of an object-oriented approach are combined to result in the following advantages: a rigid approach for identifying all objects and reusable software modules. Object operations can be modified and combined with other objects to form new computer software. This results in reduced maintenance cost and increased productivity.

3. Activity-based Costing Systems: An Electronic Industry Case

Several recent papers have emphasized the more accurate product costs reported by activity-based costing system (for example, see Cooper and Kaplan 1988a, 1988b). Firms with the following characteristics: producing a diverse mix of products, having products with short life cycle, and having high indirect cost, can benefit more from using activity-based costing systems.

To determine if the object-oriented approach is appropriate for the analysis of activity-based costing system, it is necessary to consider which data

attributes, operations, and interfaces are required for representation and retrieval. In this respect, the ABC system presents a challenge.

The concept behind ABC is that organizations perform **activities** to design, produce, distribute, sell, and service products. All these activities - engineering, manufacturing, logistics, marketing and sales, and administration - can be traced to the products that create the demand for these activities. Therefore, ABC systems break down products costs into specific activities based on the number of the **cost driver units** that the product consumes. The cost allocation mechanisms are known as cost drivers.

Activities are defined in the broadest sense to include both manufacturing processes and the myriad of actions that support the manufacturing processes. Activities include all the steps within the value chain, which includes product design, manufacturing engineering, production, distribution, marketing, and after-sale service. Activities are performed by people or by automated processes. Activity-based cost accounting, therefore, views production and support costs to be of equal importance.

An **activity center** is a group of activities that may be performed by related resources, or driven by the same cost driver. For example the activity measure for machine setup is the number of times machine setup.

This paper uses a case prepared by Foster and Gupta (1990). The company described in the case is an electronics instruments manufacturing facility that assembles and tests over 800 **products**. The firm has a traditional accounting system that includes a general ledger module. The general ledger subsystem is used to record all accounting transactions, to post them to general ledger and subledger, to prepare adjustment and closing entries, and to produce various reports including financial statements. It consists of many files such **chart of account** with account balances at various levels. The company also maintains some manufacturing cost modules that contain a **bill of material** and **a product routing** files. The bill of material (BOM) file maintains all pertinent information for products and their sub-structures of the required parts. Product routing file is used to designate the various machines, workstations, activity centers, and operational areas that a particular component passes through. The company has many **departments** such as R&D, manufacturing, engineering, marketing, and administrative department.

Exhibit 1 presents activities and cost drivers identified and used by the electronics instruments manufacturing company. Fourteen **activities** were

identified. Activities 1 to 11 cover printed circuit board assembly, while activities 12 and 14 cover the instrument build-up and testing activities. Each activity has its own **cost driver**, for example, number of pulls required for each product item is used to allocate the cost relating to activity 3: stock and pull activity.

In order to implement the activity-based costing system, in addition to the integration with the traditional general ledger system and cost accounting system, it is necessary to identify activity areas, cost drivers, quantity of driver units consumed by each activity, and the cost per unit of driver, etc. All these can be viewed as objects for the system analysis purpose. The case described in section 3 is used as an illustration for the ease of understanding the process described in the following section. The real activity-based costing system can be more complicated.

4. Object-Oriented Analysis Approach to the Activity-Based Costing System

In this section, tools and techniques related to the OOA approach include data flow diagram (DFD), entity-relationship diagram (ERD), and object dependency diagram (ODD) are used to model the ABC system.

In general, there are five steps to follow in using the OOA methodology in designing of the activity-based costing system.

1. To identify all objects and their attributes from the problem space (in this case, it is product costing).

2. To identify all operations to be performed on the objects (e.g. data updating, computing, etc.).

3. To identify all inheritance hierarchies among objects.

4. To identify all interfaces among objects.

5. To implement all objects.

Objects may be identified by using several systems analysis techniques, for example, DFD, ODD, ERD, etc.. These techniques help identifying the real world objects, attributes, and their relationships. First, we use the E-R

Exhibit 1. Activities and Cost Drivers

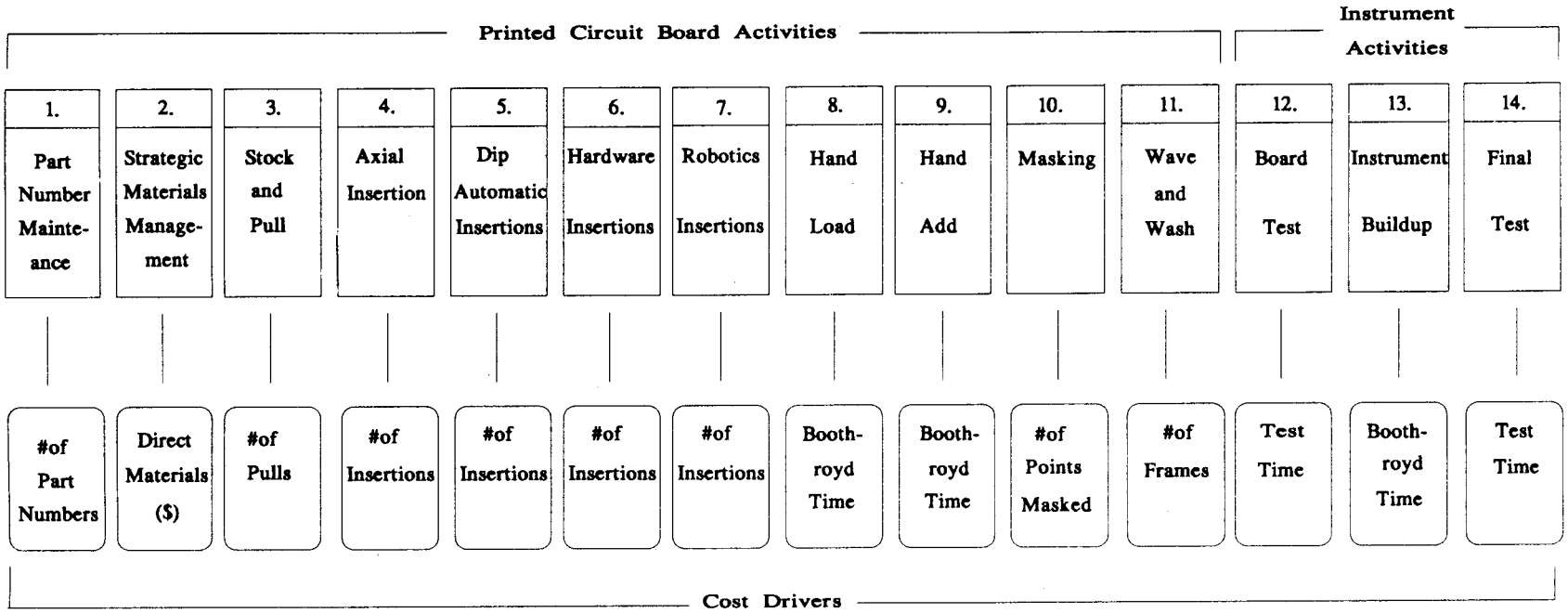


diagram as a tool to identify ABC system objects, attributes and relationships for the ABC system. Second, we use the DFD to identify operations or functions performed on objects. Third, the object dependency diagram (ODD) is then developed to present all objects, the object hierarchy, and object properties and operations.

4.1 Identifying ABC System Objects by Using an E-R Diagram

The modeling of an ABC system should be initially based on the entities identified in the ABC system, for example, the ACTIVITIES and the COST DRIVERS, etc. The object-oriented view is a natural extension of the entity-relationship view. When we think of modeling a system, we consider its objects or entities (e.g. products, activities, etc.). We therefore propose to borrow a tool, the E-R diagram, as a diagramming tool to model the external view of the ABC system objects. Gorman and Choobineh (1991) also extend the E-R diagram to the OOA approach and develop an object-oriented entity-relationship model.

Figure 2 shows an E-R diagram in identifying ABC system objects. Rectangular-shaped nodes represents entities. Diamond-shaped nodes represent relationships. We do not attempt to develop a complete model, rather the main objective is to describe the objects design process. Ten objects are identified in Figure 2. They are ACTIVITY, COST DRIVER, PRODUCT, BILL-OF-MATERIAL, PRODUCT-ROUTING, DIRECT-COST, INDIRECT-COST, CHART-OF-ACCOUNT, ABC PRODUCT-COST, and DEPARTMENT object.

A generic object form can be defined by using the object diagram notation. Figure 3 shows a diagram representation of an object. In this case, the object name is PART-MAINTENANCE. Its superclass object is ACTIVITY. Each object can be defined in the same fashion.

4.2 Identifying Operations on ABC System Objects by Using a Data Flow Diagram

The data flow diagram can be used as a tool to derive operations or functions on objects. A data flow diagram is a graphical representation of a system. There are four symbols used in a DFD as shown in Figure 4.

The bubble symbol depicts a function or a process within which incoming data flows are transformed into outgoing data flows. The rectangle

Figure 2. Identifying Objects In the Activity-Based Costing System

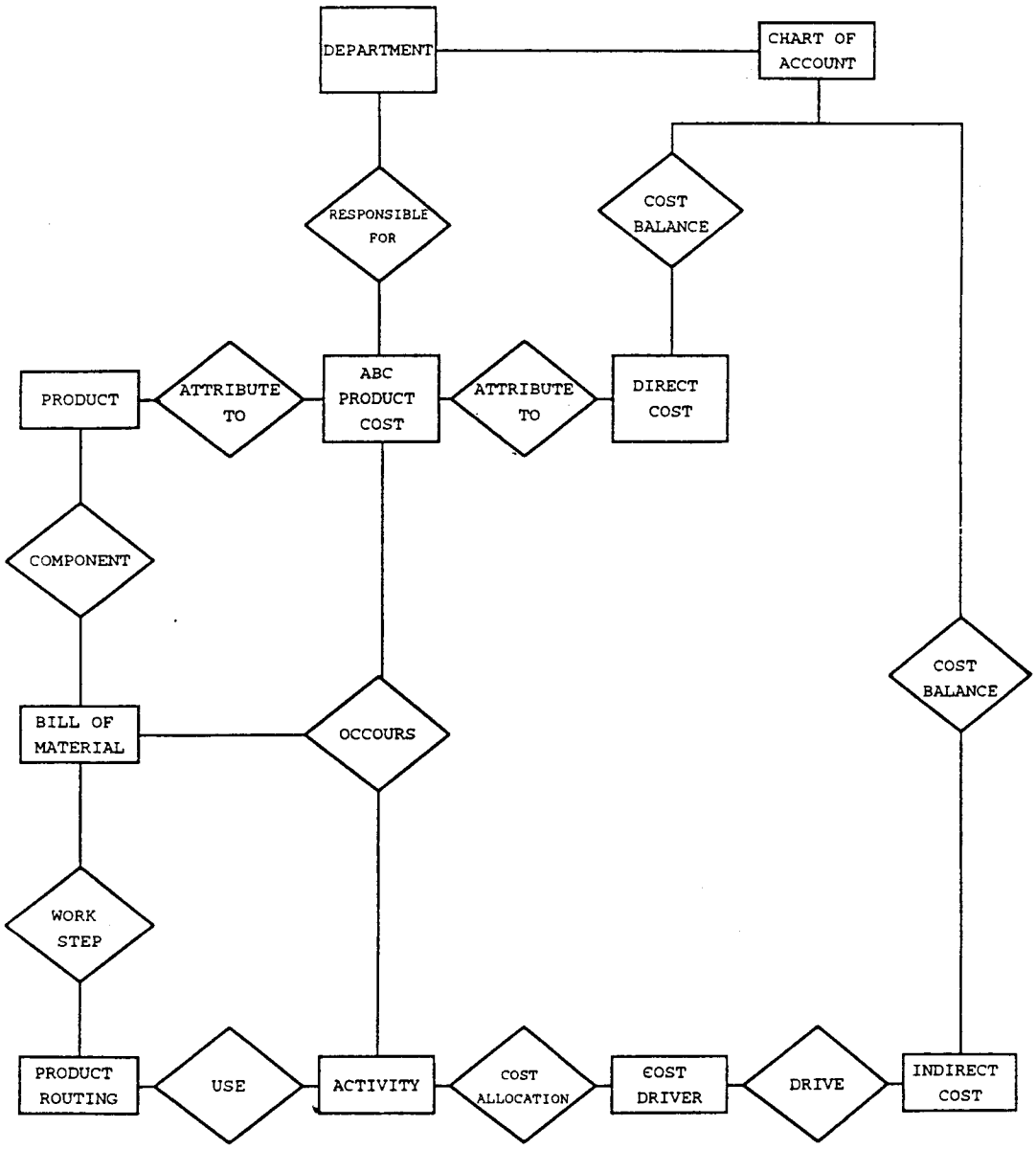


Figure 3. Defining an Object

OBJECT NAME	part-maintenance
ATTRIBUTES	activity-id activity-description
INHERITANCE	activity
OPERATION	add change delete cost allocation

symbol is used to represent an external entity which is outside of the system. The file symbol represents data store. The arrow line symbol is used to represent the flow of data.

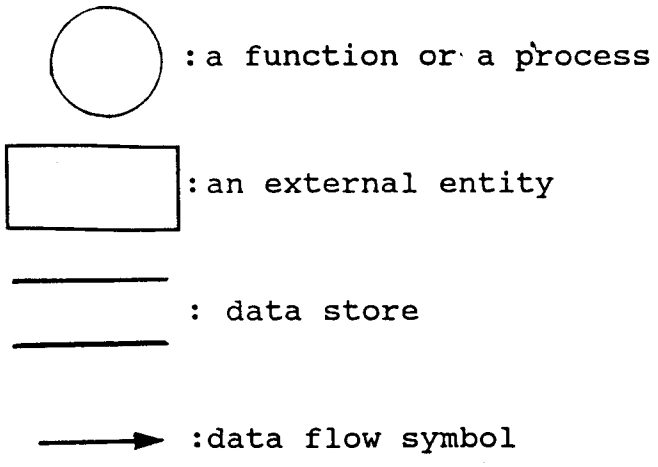
Figure 5 depicts the use of a DFD in identifying operations on the PRODUCT COST object. In the example, there are five operations or functions identified for the ABC costing system. They are DATA-IMPORT , MAINTAIN-ACTIVITY-DATA, COST-BUILD-UP, MAINTAIN-DRIVER, and PRODUCE-REPORT operations.

The DATA-IMPORT operation is used to load existing files from other accounting related files usually existing in the general ledger database as illustrated in section 3. The MAINTAIN-ACTIVITY-DATA function is used to add, change, or delete activity related data. This function is activated only by authorized approval form initiated from users. The MAINTAIN-DRIVER function is used to add, change or delete cost driver related data. The COST-BUILD-UP function is a function of calculating product costs. The PRODUCE-REPORT function is a function of producing ABC product cost for different purposes.

4.3 Activity-Based Costing System Object Dependency Diagram

The object dependency diagram is used to show the internal and external relationships among objects. It provides a formal graphical notation for modeling objects and their relationships to one another. The name of an object is listed in the top part of a box, the attributes and operations are listed in the lower part of the box.

Figure 4. Data Flow Diagram Symbols

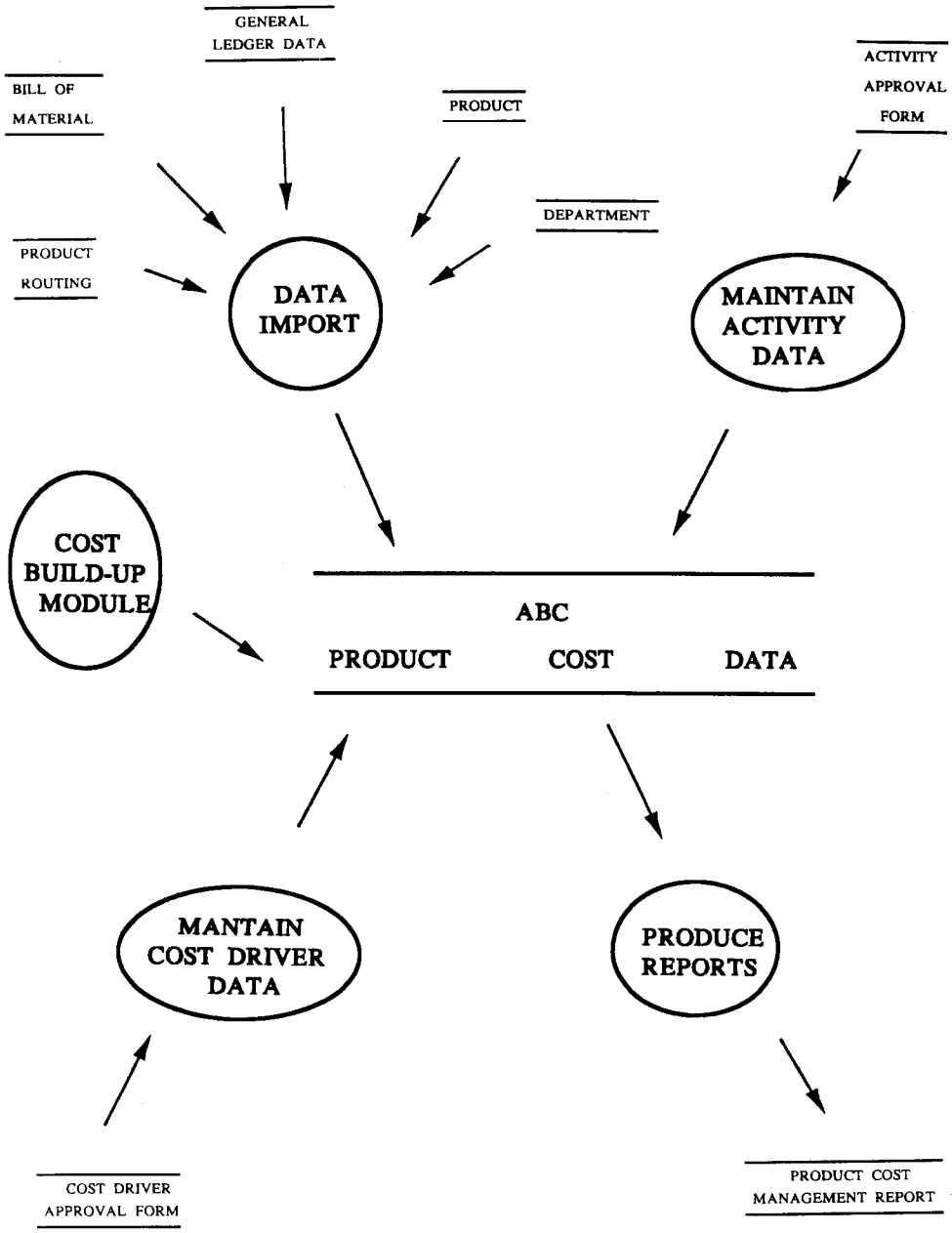


The ABC system is a subsystem of the accounting information system. The accounting information system as described in section 3 includes the general ledger subsystem, the manufacturing cost subsustem, etc.. Therefore we need the interface with general ledger database and product related database to load files (e.g., chart of accounts, account balances, product structure, etc.) to the ABC systems. A good example is the file loading operation that allows the import of some general ledger account data to ABC systems.

Figure 6 shows the object dependency diagram for the ABC system. General ledger database provides the general ledger accounts and account balances at sub-level, usually cost centers (e.g. inspection labor). Such structure allows the mapping of general ledger accounts to activities (e.g. incoming inspection).

Product-related files include product file, bill of material file, product routing file, etc.. The product file contains all products manufactured and

Figure 5. Identifying Operations on the PRODUCT COST Object



sold in the period. The bill of material (BOM) file contains all pertinent information for products and their sub-structures of the required parts. Product routing file is used to designate the various machines, workstations, activity centers, and operational areas that a particular component passes through.

Some of the input to an ABC system such as data for activity centers, machines, cost drivers, quantity, and assignment require manual entry of data. The output of the ABC system is the product cost management report for each product.

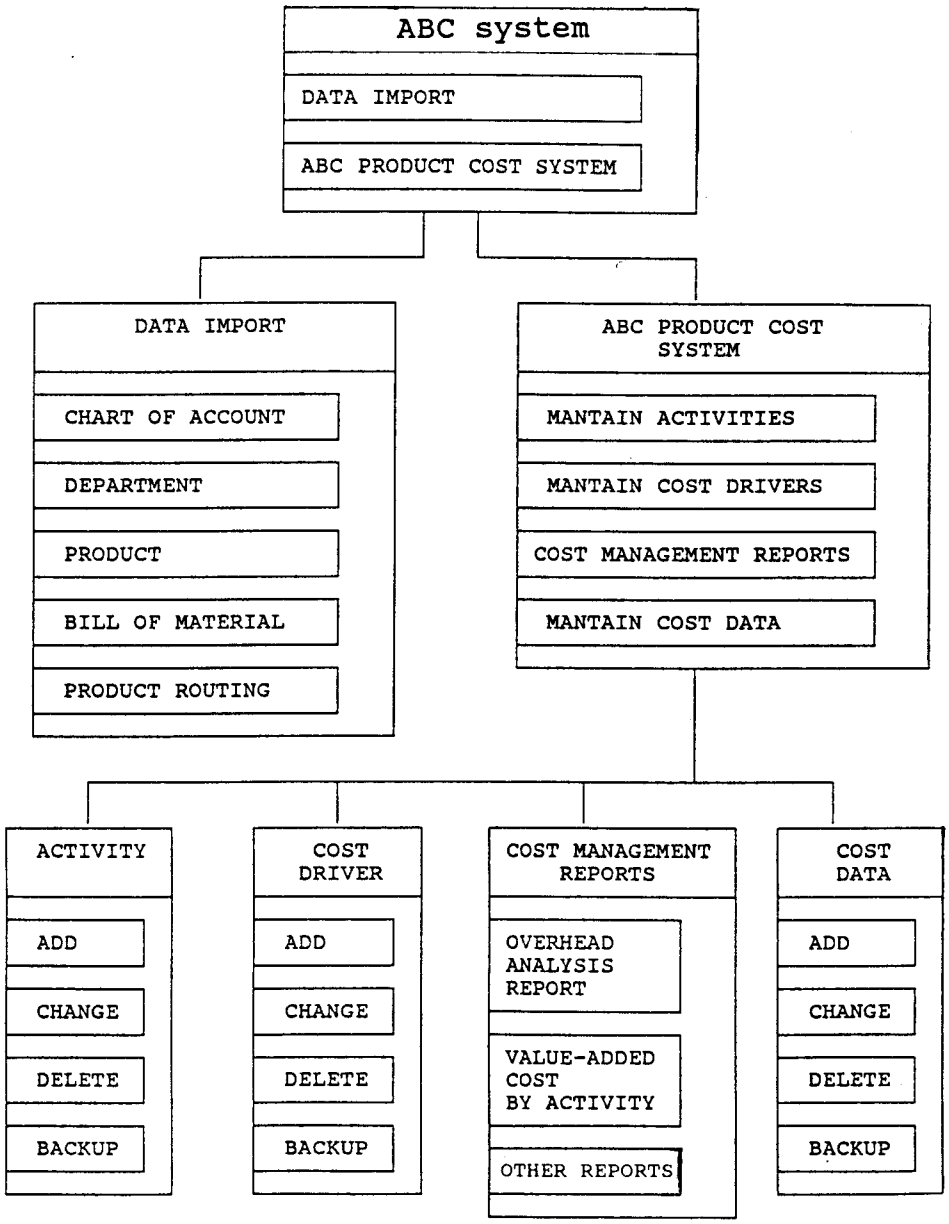
The object dependency diagram shows that many reusable modules can be interchangeable among objects. This simple demonstration presents the use of an object-oriented system analysis methodology in the activity-based costing system design process. The benefits of an OOA approach can be seen throughout the process, such as the reusability of software module, as shown in Figure 6. There are same operations performed on different ABC system objects, such as ADD, CHANGE, and DELETE. The operations and data are internal to objects. In addition, the inheritance features allows an object to inherit attributes and operation methods from other objects. Therefore, the new object needs to define only attributes or operation methods that are new or not yet defined. On the other hand, the traditional system development methods focus more on the functions rather than objects or entities. They tend to separate data and functions. Therefore, it is more difficult for the traditional methods to detect reusable modules.

5. Conclusion and Discussion

This paper attempts to integrate theories of managerial accounting and information systems, specifically in applying the object-oriented approach to help analyzing ABC system for the construction of activity-based costing system.

Activity-based costing concept is fairly new, and is not quite easy for implementation. A methodology applying the object-oriented approach to the analysis of activity-based costing systems is proposed. ABC system objects are constructed. Examples for ABC system objects and association between objects are also discussed. In addition, we demonstrate the operations required for the ABC system. The automation of activity-based

Figure 6. Object Dependency Diagram of System



cost accounting system based on the methodology proposed in this paper is expected to be easier to understand, maintain, and grow.

This paper should provide the motivation for accounting theorists to reexamine the interface between accounting and information systems theories. Research by software engineering theorists now shows that previous software development methodologies are deficient. Therefore, the object-oriented methodology theories are developed to increase software productivity and to reduce the software development cost. The object-oriented analysis and design methodology may be the direction in which software engineering will evolve in the future, so accounting theorists must be cognizant of the development in this area, particularly when attempting to integrate accounting and information systems theories. Further research should investigate the applicability of the object-oriented approach to other accounting models.

References

Brimson, J. A. 1991. Activity Accounting: An Activity-Based Costing Approach. John Wiley & Sons.

Chen, Peter. 1976. The Entity-Relationship Model-Toward a Unified View of Data. ACM Transactions on Database Systems (March): 9-36.

Cooper, Robin. 1988. The Rise of Activity-Base Costing - Part Two: When Do I Need an Activity-Based Cost System? The Journal of Cost Management for Manufacturing Industry (Fall): 41-48.

———, 1989. The Rise of Activity-Base Costing - Part Three: How Many Cost Driver Do You Need, and How Do You Select Them? The Journal of Cost Management for Manufacturing Industry (Winter): 33-46.

———, 1989. The Rise of Activity-Base Costing - Part Four: What Do Activity-Based Cost System Look Like? The Journal of Cost Management for Manufacturing Industry (Spring): 38-49.

Cooper, Robin, Kaplan, and Robert. 1988. How Cost Accounting Distort Product Cost? Management Accounting (April): 20-27.

———, 1988. Measure Cost Right: Make the Right Decision. Harvard Business Review (October): 96-103.

———, 1991. The Design of Cost Management Systems. Prentice-Hall.

DeMarco, Tom. 1979. Structured Analysis and Systems Specification. Prentice-Hall.

Eaves, B. Curtis. 1966. Operational Axiomatic Accounting Mechanics. The Accounting Review (July): 426-442.

Everest, Gordon, and Ron Weber. 1977. A Relational Approach to Accounting Models. The Accounting Review (April): 340-352.

Foster, George, and Mabendra Gupta. 1990. Activity Accounting: An Electronics Industry Implementation, Measures for Manufacturing Excellence, edited by Robert Kaplan. Harvard Bussiness Review Press.

Gorman, Kevin, and Joobin Choobineh. 1991. The Object-Oriented Entity-Relationship Model. Journal of Management Information Systems (Winter): 41-63.

Korson Tim, and D. John McGregor. 1990. Understanding Object-Oriented : A Unifying Paradigm. Communications of the ACM (September): 40-60.

Lieberman, Arthur Z., and Andrew Whinston. 1976. Design of a Multidimensional Accounting System. The Accounting Review (January): 117-145.

Rumbaugh, James, M. Blaha, W. Premerlani, F. Eddy, and W. Lorensen. 1991. Object-Oriented Modeling and Design. Prentice-Hall.

Yourdon, Edward and Larry Constanine. 1989. Structured Design: Fundamentals of a Discipline for Computer Program and Systems Design. Prentice-Hall.