University of Alberta

CAD-Based Integrated Simulation Environment (CAD-ISE)

By

Jianfei Xu ©

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of **Doctor of Philosophy**

In

Construction Engineering and Management

Department of Civil and Environmental Engineering

Edmonton, Alberta

Spring 2001

0-612-60359-8

University of Alberta

Library Release Form

Name of Author: Jianfei XU

Title of Thesis: CAD-based Integrated Simulation Environment (CAD-ISE)

Degree: Doctor of Philosophy

Year this Degree Granted: 2001

[illegible]

Jianfei Xu
7 Rehwinkel Road
Edmonton, Alberta
T6R 1Y3

Date: 1/26/2001

University of Alberta

Faculty of Graduate Studies and Research

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled **CAD-based Integrated Simulation Environment** submitted by **Jianfei Xu** in partial fulfillment of the requirements for the degree of **DOCTOR OF PHILOSOPHY** in Construction Engineering and Management.

[illegible]
Dr. S. M. AbouRizk

[illegible]
Dr. S. Alkass

A Robinson
Dr. A. Robinson

[illegible]
Dr. S. Frimpong

[illegible]
Dr. S. Alexander

[illegible]
Dr. R. Toogood

Date: 1/26/01

To My Father, My Mother,

My Father-in-law, Mother-in-law,

My Wife Yuqing, and My Sons Noah & Nick

For Their Love, Encouragement, and Support

ABSTRACT

Existing construction simulation systems lack the capability to integrate with CAD systems. Although many studies and investigations have been carried out, full integration between construction simulation and CAD has not yet been achieved. Most previous work focused on transferring data from CAD systems to simulation systems, which is limited to one-direction information flow.

This thesis undertakes an investigation of the integration of computer simulation methods with CAD to facilitate better decision-making during construction and addresses the Product Atomic Component (PAC) that is the fundamental element of integration between CAD and simulation.

The thesis objective is to conceptualize an environment that allows an engineer to specify how construction of the facility will be carried out using CAD. The environment could then populate relevant information to produce simulation models that will enable a better understanding of the construction process, better estimating of cost, and appropriate planning.

A new product-oriented simulation methodology has been developed to achieve this objective. The prototype system CAD-ISE developed based on this methodology demonstrates the feasibility and advantages of this product-oriented simulation methodology.

Keywords: CAD, Integrated simulation environment, Product-oriented simulation, Product Atomic Component, Product model, Atomic simulation model

ACKNOWLEDGEMENTS

I would like to thank my advisor Dr. S. M. AbouRizk for his direction and support throughout this research. Appreciation is also extended to Dr. S. Alkass, Dr. A. Robinson, Dr. S. Frimpong, Dr. S. Alexander, and Dr. R. Toogood for serving as members on my examining committee.

Many thanks to all members of the University of Alberta Construction Engineering and Management Group for their support and assistance throughout my graduate program. Most of all, the encouragement I received from everyone was much appreciated.

I would like to express my appreciation to North American Construction Group and the people in this company for their understanding and support.

I would like to thank my technical editor, Cam Fraser, who has helped to make this thesis better.

I thank God for my wife, Yuqing, whose love, patience, and understanding I will always need. I am thankful for my sons, Noah and Nick, and the joy they bring.

I would like to recognize my parents for their continuous support and encouragement throughout my university education.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ACMM – Atomic Construction Method Model.

ASA – Atomic Simulation Activity.

CAD – Computer-Aided Design.

CAD-ISE – CAD-based Integrated Simulation Environment.

CMSL – Construction Method Simulation Library.

EM – Environment Model.

GIS – Geographical Information System.

PAC – Product Atomic Component.

PH – Product Hierarchy.

PM – Product Model.

PN – Product Network.

POS – Product-Oriented Simulation.

RM – Resource Model.

SM – Simulation Model.

WBS – Work Breakdown Structure.

CHAPTER 1 INTRODUCTION

1.1 Introduction

Computer Integrated Construction Simulation (CICS) is an advanced technology that meets the needs of management in construction. The objective of CICS is to improve the efficiency, accuracy, and prediction ability of management by collecting all types of information associated with the production, simulating the different construction scenarios, and making information readily available to all participants. The integrated information allows the survey, GIS, design data, and simulation results to be used for construction planning, scheduling, estimating, and animation. It also provides downstream data for facility management.

Simulation provides an efficient and powerful means of analyzing construction operations. The dynamic nature of resource flow and uncertainties in process duration can be realistically modeled and analyzed using simulation. With a better understanding of the construction process before actual work begins, construction engineers can adjust resource allocation to optimize utilization, eliminate bottlenecks, increase productivity, and reduce costs. These benefits and others are attracting industry leaders to CICS.

Computer-Aided Design (CAD) systems, such as AutoCAD and Microstation, are considered essential in today's engineering and design industry, and are widely used in these fields. The use of 3D CAD systems in the detailed design phase of projects is extensive and rapidly increasing. Unfortunately, they are seldom used in the construction phase despite the fact that a large portion of the total project cost is usually incurred during this phase. When computer applications are used in the construction phase (such as simulation, scheduling, estimating, cost control, and animation), they are largely fragmented and do not make use of all of the information available from the 3D CAD system and others applications; in other words, the programs are not integrated. With advanced technologies, such as Object-Oriented Programming (OOP), Object-Oriented Design (OOD), and Object-Oriented Simulation (OOS), construction firms have an opportunity to develop an integrated CAD application to increase productivity and accuracy, and improve system integration in construction management.

Simulation can be a powerful tool for construction management, especially for comparison of construction methods, allocation of resources, scheduling, and estimating. It provides advantages in modeling repetitive construction operations, such as earthmoving, dam building, paving, and road construction. In most cases, this type of construction is associated with equipment intensive operations, and simulation can play an important role in planning, scheduling, selecting resources, etc. Alternative solutions to a problem can be studied and compared, which gives the investigator insight into the operations of the system and its components.

1.2 Background

1.2.1 Limitations of Current Systems

1.2.1.1 3D CAD systems

Although 3D CAD systems are extremely powerful, they are not being utilized thoroughly in the architecture, engineering, and construction industries. They still serve only as automated drafting tools, which are limited to their own narrow areas of specification rather than creating product objects that could later be utilized by other applications, such as simulation.

Current CAD systems have inherent shortcomings, which diminish their feasibility for architectural practices. Almost all conventional CAD systems rely on a pure geometric data model requiring all non-geometric information about objects of architectural interest to be attached to these geometric entities. This restricts their ability to describe semantically-dependent relationships.

1.2.1.2 Product Modeling Systems

Many product-modeling systems have been developed during the past decade, such as ID'EST (I. Kim et al. 1997). Most of these systems focus on ways to extract and transfer the information from the CAD electronic drawing to other applications to avoid re-entering data. After completing data extraction and transfer, the link between product model and data is immediately destroyed. Some product modeling systems, such as ISICAD (N. A. Kartam 1994), provide the user with the opportunity to build semantic network relationships (SNR) based on the product components that form the Product Hierarchy structure. However, the product components created in these systems are only groups of CAD graphic primitive objects, such as lines and rectangles, rather than the real product objects. Advanced product modeling systems attempt to extend current CAD systems by adding plug-in tools, which aid designers in creating the product object during the design phase by using a built-in product model library. Though this methodology could be useful for the building project, it is not as useful for heavy civil projects, since there is no way to pre-define the product model to build the product model library.

Current product modeling systems lack the capability to create product models that can act as the information container or knowledge carrier for simulation purposes. The product models created by the current modeling systems also fail to serve as the integrated entities that are created during the design phase, conduct the simulation, and collect and distribute the information in the course of the life cycle.

1.2.1.3 Simulation Systems

Although construction simulation has been studied for many years in academic research, it has not been adopted and accepted in industry to any significant degree. The major limitations to its use by industry are:

- The increased complexity of simulation models.
- The large amount of information that must be included in the simulation model.
- Its inability to intelligently store and manipulate the vast majority of the information relevant to the construction process.
- Its inadequacy in handling construction-related entities.
- Its inability to integrate with other key elements of the process, such as design.

1.2.2 Needs of the Integrated Product Model

A large amount of information in various formats is created and used throughout the life cycle of a construction project. This information is used by many participants at different times, including the design phase, plan stage, and construction phase. To take advantage of this huge amount of information, an information repository or integrated product model must be created at the beginning of the project to collect and queue the data as the project moves through its life cycle.

Detailed simulation of construction requires a great deal of information, most of which can be derived from the CAD drawing. Thus, for simulation in construction to be accepted by the industry, it must provide a simulation environment that is fully automated through integrating with CAD systems. However, conventional CAD and construction simulation technology do not allow for the automated derivation of simulation object representations from general design object representation. First, product design representations in a general design environment (e.g. a CAD tool) do not provide information for the product object needed for simulation. Second, such representations do not allow for conducting domain-relevant operations such as grid generation or atomic component definition. Evidently, conventional CAD tools are not able to create objects that meet the simulation requirements. The integrated product model, which allows for the derivation of a simulation object representation from the design object representation, must be addressed. This will allow the construction management team to take advantage of the CAD files received from the design team to build an information management framework for simulating, planning, and managing construction tasks.

1.2.3 The Potential of the Integrated Product Model

Although CAD has improved significantly, it has had relatively little impact on the improvement of construction planning and management. The potential for cost savings is usually greater in construction than in design; CAD, with its ability to model 3D space, should therefore become more of a construction aid. The integrated product model generated from 3D CAD can be more effectively used to integrate different construction applications.

The integrated product model can easily achieve the primary aim of integrating designs using a CAD environment with simulation, and can subsequently be extended to incorporate project control applications. Using an integrated repository of data, the framework provides a means to achieve an effective and coordinated exchange of information within the inter-disciplinary processes in design management. The information contributed by each practitioner is stored consistently, and made accessible to other project-associated practitioners through the integrated product model.

Effective communication is essential in all aspects of a project at any phase, from design to construction and operation. Failure to convey the right message between practitioners can often result in misunderstanding project alternatives and objectives. Failure to exchange construction information can cause delays, error, cost overrun, accidents, and structural catastrophes. The integrated product model provides a solution to these problems by accurately transferring information between these practitioners.

Animation is an increasingly popular technique that can be used to verify, debug, and validate simulation models. Further, it can also show, in an easy-to-understand manner, the insights gleaned from the underlying model, thus significantly increasing its credibility. Recent advances in computer animation can be attributed to the increased popularity of simulation modeling. Typically, the animation of a simulation model involves a graphical computer application that uses changes in the position, shape, or color of icon to illustrate changes in the state of the simulated system. The integrated product model that acts as the knowledge carrier makes this possible.

1.3 Thesis Objectives and Scope

1.3.1 Problem Identification

The information built into the 3D CAD electronic drawing, such as geometry and position, and the visualization capabilities of the 3D CAD models are underused by construction management teams for construction simulation, planning, and estimating. The gap between design and construction in terms of 3D CAD information makes industry resistant to accepting simulation. Failure to integrate physical design components with simulation entities, when extended to the scheduling work packages and estimating work packages, creates obstacles for other applications in construction, such as scheduling and estimating.

To use the information stored in the 3D CAD drawing, many researchers have endeavored to transfer and link data between CAD and other application systems. However, file types such as DXF, GKS, PCX, and IGES, which are used by popular CAD systems, are intended to facilitate the use of the model by computer graphic applications and therefore are rarely used by construction applications.

1.3.2 Contributions

The implementation of this methodology is expected to make the following major contributions:

1. The extractor model presented in this thesis is the first research effort aimed at generating the Product Atomic Component (PAC) from a CAD drawing. All future information associated with this atomic production component will be attached at different stages during the construction period. The Product Hierarchy

and Product Network provide an approach to visually and practically create a product structure and construction sequence.

2. The research provides a system that collects data throughout the course of product construction, from the early stages of design until the completion of construction. As the process continues, more information will be brought into the system; information must only be entered once, regardless of where it is entered.
3. All models in the CAD-ISE contribute to and draw from the product object. It is believed that the CAD-ISE will have significant advantages over the existing piece-meal-style simulation programs. Using an integrated repository of data, the framework provides a means to achieve an effective and coordinated exchange of information within the inter-disciplinary processes in design-management.
4. The product-oriented simulation methodology developed in this thesis offers a new means to conduct construction simulation.

The research is expected to simplify systems through: a) automated data entry from CAD, which also increases the accuracy; b) a construction process simulation model, which allows users to select the model by relying on familiar construction methods.

1.3.3 Thesis Objectives and Scope

The main objective of this research is the development of a customized CAD environment capable of generating an object-oriented product model to achieve the integration of 3D CAD with construction simulation. Through this environment, design-related data is captured and stored within product models. These object-oriented product models provide necessary input data to computer-based construction applications that perform simulation, scheduling, estimating, and cost control. Thus, it serves as a unified medium for integrating design and construction. The four major goals that make up the overall objective are described below:

1. Develop the framework of an object-oriented product model for integrating 3D CAD design, simulation, and other construction applications.
2. Conceptualize a product-oriented simulation methodology that focuses on the product, the simulated subject, instead of the processes or activities.
3. Develop an extended CAD environment based on AutoCAD. This environment will serve as the user interface for the extraction of information from CAD drawings, entering extra information for simulation, conducting the simulation, and displaying the simulation results.
4. Provide an extended application prototype that uses the information in the product model.

The scope of the prototype system design will focus on heavy construction operations, such as earthmoving, dam and road building, and pipeline construction, which are characterized by homogeneous products and repetitive processes. Major efforts will concentrate on developing the prototype system modules and the methodology for developing the product model. The classic simulation model, earthmoving, will be used to prove the methodology and the prototype system.

The focus of the system is on the construction phase of the project from the viewpoint of a construction management team performing the simulation, planning, estimating, and cost control of a construction project. The 3D CAD product model generated from the information provided by the CAD drawing and other applications acts as an initial input for the system.

The Construction Simulation Method Library will provide enough construction methods, processes, and resources to support the system. The information that is embedded in the integrated product model can be extensively used in construction applications, including scheduling, estimating, and animating.

1.4 Research Methodology

To accomplish the proposed research objectives, the methodology, as well as a prototype system, will be designed and developed to integrate the stand-alone 3D CAD system and simulation model to conduct the simulation under an extended CAD environment. The CAD-ISE will use C++ as its developing host language, and ObjectARX, an AutoCAD run-time extension Software Development Kit (SDK), will be used to implement the extractor and environment model, as these two models work closely with AutoCAD. An object-oriented method will be employed throughout the system, and all modules will be reusable classes. The main advantage of the object-oriented approach is that it naturally matches the realistic representation of real world objects. The product model developed in the research is modeled on the form of objects that represent and capture the shape, structure, and behavior of real world entities, making them ideal for simulation purposes. The entire system will run on the AutoCAD platform, thus, CAD-ISE can take advantage of AutoCAD's powerful graphic function and seamlessly integrate CAD with simulation.

The main tasks in the proposed research will be:

1. Survey and study the current product model.

2. Create a conceptual design for the product model, including identification of the model attributes and behavior.
3. Create a conceptual design for the simulation environment under the CAD environment, including the extractor model, which is responsible for the generation of production components. Also included is the link model, which serves as a simulation process editor to link the construction method simulation foundation and fundamental simulation engine with the product model in order to conduct the simulation.
4. Develop a prototype system to demonstrate the proposed methodology.

1.5 Thesis Organization

The thesis is organized into eight chapters detailing the design, development, and case study of a prototype system for integrating CAD and simulation. Chapter 1 describes the background of this research study, defines the goals to be accomplished, identifies the problem to be solved, defines the scope of the prototype system, and discusses the methodology used in conducting this research study. Chapter 2 summarizes the literature review. Chapter 3 provides an outline of the entire CAD-based integrated simulation environment. Chapter 4, which presents a conceptual design of the product-oriented simulation methodology used by CAD-ISE, describes the frame of the product-oriented simulation methodology and provides the prototype system architecture. Chapter 5 describes the actual development of the prototype system and discusses the object class and prototype system features. Chapter 6 uses a classic simulation case to prove the feasibility of the product-oriented simulation methodology and describes the detailed procedure to conduct simulation under CAD-ISE. Chapter 7 provides a case study to test and evaluate the prototype system. Chapter 8 concludes the research results, reviews the achievement of the stated goals, discusses their limitations, and makes recommendations for future study.

CHAPTER 2 LITERATURE REVIEW

2.1 Introduction

This chapter summarizes the research that has been done with respect to the integration of CAD and construction simulation. First, the studies of integration of CAD and product modeling are discussed. Second, most construction simulation systems and tools are identified. Third, the limitations of current simulation modeling are recognized. Finally, the CAD-based Integrated Simulation Environment (CAD-ISE) developed in this thesis is briefly discussed.

2.2 CAD Integration

The need for design-construction integration has attracted many researchers and large firms who study the methodology and develop computer software for this purpose. With regard to CAD integration, this research can be summarized as follows: 1) Link, building the linkage between CAD data and the affiliated applications; 2) Transfer, passing the necessary information to these programs; and 3) Generator, creating the product object during the design stage using the developed CAD tools. As for the production model, major research concentrates on 1) Standard for the Exchange of Product Model Data (STEP); 2) Hierarchical Structure; and 3) Relationship Model (K. Ito 1998).

2.2.1 Integrating with CAD

P. V. Dharwadkar and T. M. Gatton addressed an object-oriented building model for CAD/schedule integration. The initial schedule can be generated automatically from 3D CAD data by using the JSpace object-oriented environment developed by Jacobus Technology Inc. The overall approach for developing the project model is to represent the construction project in terms of objects. Three major classes are presented to achieve the goal: the building component class is used to create a hierarchical building object model; the construction activity information class is modeled using the concepts of work packaging and the Work Breakdown Structure (WBS); and the construction information class is used to represent the standard construction methods used for the construction of building projects (P. V. Dharwadkar et al. 1995).

The Construction Simulation Tool Kit (CST) was introduced by J.F. Skolnick (1993). The CST links a construction schedule and a 3D CAD model of a construction project to produce a simulation of the construction process. The CST allows the user to link project activity data from a project management system to objects in a 3D computer model of the construction project. This information is then used to produce a custom simulation of the construction process. The user can then simulate multiple alternatives and views simultaneously. In CST, the model should be divided into a number of parts or "objects" that can be referred to independently. Once these models have been developed, they can be exported into an ASCII file format that can then be imported into the CST's database using a translator. After loading the project schedule and model data into CST's database, the user can begin creating relationships between the two; in effect, it is a preliminary, schedule-based walk-through rather than a full simulation.

CIPROS (I.D. Tommelein et al. 1994) uses a modular representation to create discrete-event simulation networks, and to aid in relating simulation output back to the design and construction plan of the facility to be built. CIPROS users must identify and describe attributes of components to be constructed based on the facility's design drawings and specifications, and they must develop a Critical Plan Method (CPM). They must also select a construction method to perform each activity by retrieving the appropriate elemental simulation network from a library of networks that represent such methods. CIPROS then pieces together the networks based on sequential relationships from the plan, and property values taken from the drawing and specification data (I.D. Tommelein et al. 1994).

Builder (J. Cherneff et al. 1991) combines a computer-aided design system with knowledge-based programming and database techniques to provide the following features: 1) use of a semantic network of objects as a common language for representing and communicating an architecture-engineering-construction problem state; and 2) a modular and practical approach to knowledge-based construction planning, where each building-component data type contains the rules and procedures required to design and schedule it (J. Cherneff et al. 1991). The Draw KM presented in Builder is a knowledge-enhanced CAD system that provides a front end for the input of architectural design information. As the designer builds the drawing with Draw KM, the Draw model will work in the background to create the semantic network representation of the objects. It then serves as an interpreter for the Planner, the other model in the Builder, by translating the relevant features of the drawing into a declarative representation, i.e., the semantic network. The major contribution of the Builder system is the consistent object-oriented representation of construction components throughout the process, from the drawing utility to network creation and estimating. The important issue addressed by this system is the integration of CAD design with construction planning. The use of 2D CAD drawings rather than 3D CAD models in the Builder is the major limitation of this system, as is the requirement that all CAD drawings be produced by Draw KM, provided by the Builder.

To take advantage of the increasingly electronic and object-based descriptions of design, schedules, and estimates, Martin A. Fisher et al. (1996) addressed the integration mechanisms that translate design descriptions into schedule and cost views of projects. Three models are discussed:

1. Product model (what);
2. Process model (when);
3. Construction model (how).

The system generates activities and their sequential relationships by matching construction method templates with higher-level activities. The activities are linked to building components in the product model generated from the CAD drawing. The construction methods model supports the component- and activity-based elaboration of a process model. The system links design information to construction planning and scheduling through the integration of the three models: the process model, which selects the construction method from the construction methods model, applies to building components in the product model. The major contribution of this research is the computer-interpretable construction method models that provide links between design description, cost estimates, and schedules.

ISICAD (Interactive System for Integrating CAD and computer-based construction systems, N. A. Kartam 1994) captures selected design data and represents this data in an object-oriented project model during the design stage of AEC projects. By inheriting the design data in the hierarchy of object-oriented classes and sub-classes, all functional, geometric, structural, construction, and design evolution data from the different design domains are shared across class boundaries. These domains include architectural, engineering, constructural, estimation, management, and scheduling. This design data can be used as input to various computer-based construction systems to support reasoning about geometric, topologic, spatial, and temporal issues in the project life cycle. During the design stage, design data was captured through the creation of a design library, which was built by BLOCK. BLOCK groups individual CAD primitives with lists of attributes of design elements, which were stored within the block of the element and linked to non-graphic characteristics of the elements.

2.2.2 Production Modeling

I. Faraj and M. Alshawi (1996) defined product models as software representations that support a project throughout its life cycle, from design and construction to the maintenance of the project. A product model is therefore a populated instance of the product data model.

Integrated Design Environment using STEP methodology or ID'EST (I. Kim et al. 1997) uses a core data model to provide semantically meaningful descriptions of buildings. The suggested core data model provides the basis for consistent design data throughout the design and construction process. By means of the core data model, integrated CAD systems can represent and exchange design information at a semantic level, such as exchanging components and features of a building, rather than graphical primitives. Consequently, the core data model reduces misunderstanding and communication problems among the multiple disciplines of architectural design. ID'EST comprises Prototype Architectural Data Model (PADM), developed according to the methodology of STEP, as its fundamental part. PADM defines the structure of the core data model, as well as the syntax of the exchange format. The data is exchanged between CAD and STEP through the STEP/DXF interface, which makes use of user-defined mapping tables.

Mather (1998) described a model in his thesis: "A CAD-based Simulation for Construction" that incorporates the site and design information from CAD, and uses this data to define the simulation model from predefined atomic models and link this information to simulation model activities and components.

For construction planning, especially lifting device planning, Kenji Ito (1998) described an object-oriented building model that includes three-dimensional topological information such as type, shape, and material of building elements with its weight, center of gravity, and its relationship between the building elements.

2.3 Simulation

Simulation can be classified as either discrete or continuous according to the rate of change for the simulated system. In most simulation, time is the major independent variable, and dependant variables in the simulation are functions of time. In *continuous simulation*, the dependent variables of the model may change continuously over simulated time. *Discrete simulation* occurs when the dependent variables change discretely at specified points in simulation time, referred to as event times (Pritsker

1997). In discrete simulation, it is assumed that the state of a system changes instantaneously at specific times marked by events. Most construction processes can be effectively modeled using discrete simulation.

2.3.1 Simulation Systems and Tools for Construction

Following the early work of Teicholz (1963) and Gaarslev (1969) in the field of construction simulation, Halpin (1973) developed the Cyclic Operation Network (CYCLONE) methodology. CYCLONE uses a graphical modeling format to model work states and the flow of entities through the system, facilitating ease-of-use for modeling construction operations. The development of CYCLONE inspired a number of other simulation systems based on the same modeling methodology.

RESQUE (Chang 1986) was designed as a significant enhancement to CYCLONE, not limiting the model to the information conveyed by the network. In addition to the CYCLONE network, a RESQUE model has an overlay that defines resource distinctions and increases simulation control. The overlay follows a Process Description Language (PDL) specific to RESQUE in an effort to overcome the resource characterization capabilities missing in CYCLONE. The solution presented by RESQUE through PDL is a significant improvement over CYCLONE insofar as recognizing distinctions among resources that flow through the same path.

The COOPS construction simulation system (Liu 1991) is an extension to CYCLONE designed and implemented using an OOP. The simulation network is a collection of objects such as activities, queues, and links that are drawn interactively on the screen. These objects perform the simulation by reacting to messages sent from other objects. COOPS's interactive graphical model definition is a great improvement over previous construction simulation systems: modeling elements are picked, placed, and moved directly on the screen, and there is no need to enter a textual equivalent of the network.

CIPROS (Odeh 1992) is both a process- and project-level planning tool. It contains an expandable knowledge base of construction techniques and methods, and makes ample use of a hierarchical object-oriented representation for resources and their properties. CIPROS extends its resource characterization capabilities beyond RESQUE by allowing resources to possess multiple real properties as well as more complex resource selection schemes. It integrates process- and project-level planning by representing activities through process networks, all of which can use a common resource pool (Martinez 1996).

DISCO (Huang and Halpin 1994) is pre-processor and post-processor to Micro-CYCLONE. The pre-processor is an interactive graphical user interface similar to COOPS. The post-processor animates a simulation by "playing back" various statistics as they occurred during the simulation.

STROBOSCOPE (State and Resource Based Simulation of Construction Processes), developed by Martinez and Ioannou(1996), combined activity-scanning with a powerful programming language. This approach and the use of characterized resources in a stochastic environment make typical engineering calculations for heavy equipment performance relatively easy to implement and significantly more accurate.

Tommelein and Odeh (1994) developed an interactive object-oriented simulation system that uses the concept of hierarchy and matching resource properties to model construction operations. Oloufa (1993) used object-oriented programming to simulate a simple earthmoving operation. Sawhney and AbouRizk (1994) developed a computerized tool for hierarchical simulation modeling using object-oriented concepts and event driven programming. The Resource Based Modeling (RBM) approach, developed by Shi and AbouRizk (1997), defines basic processes of resources as atomic models and stores them in a model library. Special Purpose Simulation (SPS) was addressed by Hajjar and AbouRizk (1997) through research on CRUISER, CSD, and AP2-earth.

2.3.2 Limitations of Current Simulation Modeling

Although construction simulation has been studied in academic research since it was first popularized by CYCLONE (1977), it has not been adopted by industry to any significant degree. Most research on construction simulation has been done in the area of process-interaction simulation, which focuses primarily on construction processes or activities. This approach creates a gap between the product, the simulated object, and the simulation model. The major limitations of this process-interaction simulation methodology are due to this gap.

2.3.2.1 Data Preparation and Acquisition

Modeling a system is essentially a means of describing it accurately, a process in which data plays a fundamental role. The current simulation methodologies fail at the construction project level because of the increased complexity of simulation models and the large amount of information that must be included. Though a large amount of data must be prepared and entered into the simulation system, this data is often available from other sources, or can be generated from the existing data. In other words, current simulation systems have failed to take advantage of the data available in existing files, such as CAD design drawings.

Data acquisition is the process of obtaining data on a phenomenon of interest. In most cases, data acquisition involves the use of historical data from a similar project completed previous data collected for a current project will benefit coming projects, as well as itself. However, the unique environment and variable conditions under which construction projects take place cause great difficulty in accurately calculating factors such as productivity using information from a previous project. The current simulation systems fail to offer a means of collecting data and using it to adjust the simulation system's factors, which would allow the simulation to be run at any given point to produce more accurate predictions for the on-going project.

2.3.2.2 Storing and Manipulating Information

Current simulation methodologies are constrained by an underlying design that limits their extendibility. Further, they lack the ability to intelligently store and manipulate the vast majority of information relevant to the construction process (AbouRizk 1997). Simulation is a planning tool that should link the design and construction stages; as information flows from the design stage to the construction stage, more information is added, and it becomes critical to store and manipulate large amounts of useful information. Existing simulation methodologies cannot accommodate this stage of information flow due to the underlying limitation of their design.

The information carried from the design stage, the information generated during the simulation, and the information collected while the project is under construction, require an intelligent depository which not only exists within the simulation, but also exists before and after the simulation. The activity cannot take charge of this as it only takes place in a short period. Since the most current simulation systems in construction are activity-based, which focus on the activity while modeling, the intelligent depository of information has not been addressed. As a result, it hinders the application of simulation to the construction industry.

2.3.2.3 Simulation Knowledge

The increased complexity of simulation models results in a need for highly educated users who are well versed in simulation knowledge. The difficulties involved in modeling a construction process using the defined modeling elements (e.g. Queue, Normal, and Combi nodes in CYCLONE) have been widely experienced by all levels of users, from academics to construction engineers. Modeling the construction process with the current simulation systems requires simulation knowledge that takes months or even years of learning, training, and experience. It has been recognized that the difficulty in its use has greatly hindered the application of simulation in the construction industry (Shi and AbouRizk 1997). Although it was concluded in a workshop sponsored by the National Science Foundation twelve years ago that more research needs to be done to make simulation an easy-to-use tool for the practitioner (Ibbs 1987), no substantial advancement has been reported (Shi 1999).

2.3.2.4 Complexities of the Construction Site

Complexities in construction generally involve taking into consideration factors such as site condition, site layout, location, internal constraints, and physical information (e.g. type and geometry). These factors, in turn, affect the construction methods that would be employed in the construction, and the resources that would be involved in those methods. In other words, the simulation model should provide a way to address these factors; otherwise, the model will be inaccurate.

Taking earthmoving as an example, the source contains three types of soil: clay, sand, and rock. Different construction methods and equipment are required to excavate each soil type. The issue that arises from this situation is when and where the different soil types are going to be met during excavation, and how to specify it in the simulation model. Since this information links to the site surface – including original, sub, and final surface – the simulation system should provide a means to access the CAD drawings to produce a solution to this issue. However, current simulation fails to do so.

2.3.2.5 Integration

In order for simulation in construction to be successful, it must provide an integrated solution for the construction practitioner. Past experiences with a number of construction companies indicate that, unless the provided computer tool supports estimating and scheduling functions, it will not be utilized (AbouRizk 1997). Obviously, this indicates that one of the reasons for the failed acceptance of simulation in construction industry is its inability to integrate with other key systems, such as CPM, estimating, and animation, in order to generate more meaningful reports for the construction engineer.

Due to the lack of integrated post-process systems such as animation, it also creates difficulty in building and verifying the simulation model, which is extremely time consuming. It has been shown that the time and effort required to build a simulation model for a construction project does not provide a positive return on investment (Shi and

AbouRizk 1994). Additionally, the large amount of data entry and preparation cause many construction engineers, who intend to use simulation, to hesitate because of the lack of integration between the CAD systems and current simulation packages, an integration that would extract and translate the information from the CAD system.

2.4 CAD-Based Integrated Simulation Environment (CAD-ISE)

This thesis attempts to investigate an alternative construction simulation methodology from a product-oriented perspective. This methodology focuses on the product, the simulated object, instead of the processes or activities. It specifically addresses construction needs by providing a flexible and open architecture that enables: 1) the detailed and efficient modeling of a complex system, 2) the efficient development of special purpose simulation tools for the construction industry, and 3) the simplifying of the simulation modeling process through integration.

The CAD-based integrated simulation environment applies these concepts in order to provide a simulation environment that integrates with CAD, requires minimal simulation knowledge, and is simple to use. Utilizing CAD-ISE as a modeling environment, an engineer will be able to develop a model of the construction process for surface mine preparation, an earth-filled dam, and linear projects such as road and pipeline construction. The model can then be experimented with to study alternative construction methods, allocate resources, plan construction, estimate time and cost, and evaluate constructability issues.

The extractor model presented in CAD-ISE is the first research effort aimed at generating the Product Atomic Component (PAC) from the CAD drawing. All future information associated with this atomic production component will be attached at different stages during the construction period. The Product Hierarchy and Product Network provide an approach to visually and practically create a product structure and construction sequence. CAD-ISE collects data throughout the course of product construction, from the beginning of design until the completion of construction. As the process continues, more information will be brought into the system; information must only be entered once, regardless of where it is entered. All models in the CAD-ISE contribute to and draw from the product object. It is believed that the CAD-ISE has significant advantages over the existing piece-meal-style simulation programs. Using an integrated repository of data, the framework provides a means to achieve an effective and coordinated exchange of information within the inter-disciplinary processes in design-management. The product-oriented simulation methodology developed to integrate CAD with simulation in this thesis offers a new means to conduct construction simulation. The CAD-ISE is expected to simplify systems through: a) automated data entry from CAD, which also increases the accuracy; and b) a construction process simulation model, which allows users to pick up the model by relying on their knowledge of familiar construction methods.

CHAPTER 3 CAD-BASED INTEGRATED SIMULATION (CAD-ISE)

3.1 Introduction

The developed computer simulation system should be accurate, robust, easy to use, provide a natural modeling environment resembling to the greatest possible extent the construction operation itself, and integrate with design, estimating, and scheduling functions.

Although construction simulation has been successful in academic research, it has not been accepted into the construction industry to any significant degree. The development and verification of these models are extremely time-consuming and require intimate knowledge of both simulation and the construction operations being modeled. It has been shown that the time and effort required in building a simulation model for a construction project does not provide a positive return on investment (Shi and AbouRizk 1994).

For simulation in construction to be accepted by the industry, it must provide a simulation environment that is fully automated, requires minimal simulation knowledge, is simple to use, and provides quick results. In the construction business, most contractors specialize in a specific area of expertise; logically, they are only interested in simulation tools that are specific to their own business. Special purpose simulation tools for a specific area of construction can be developed to meet the needs of these contractors. These tools can be presented in a simple and graphical context, require minimal simulation background, and provide quick results (AbouRizk and Hajjar 1997).

Although a fair bit of research on CAD simulation has been done to date, the majority of the work is focused on animation. This thesis presents an approach that truly integrates CAD and discrete event simulation, rather than simply producing animation or extracting data from CAD and transferring it to the simulation model.

The proposed research will focus on developing a fully CAD-based integrated simulation environment for special construction projects, such as earth moving, dam building, paving, and road construction. This environment will serve as the platform that will enable construction engineers or project managers to quickly and effectively develop simulation models and use the simulation to assist them in decision-making. Fig. 3-1 illustrates the CAD-based integrated simulation architecture.

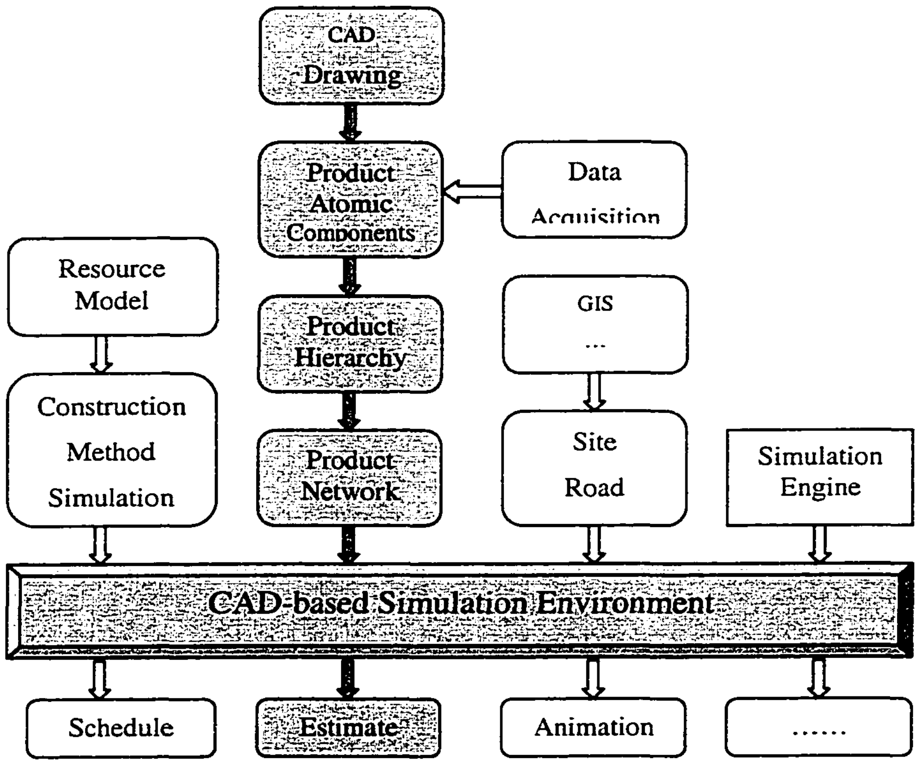


Fig. 3-1 CAD-ISE Architecture

3.2 Product Model

Product modeling, which plays a critical role in CAD-ISE, should be viewed as a process leading to an information model that provides an abstract description of facts, concepts, or instructions about a product or a set of products (ISO C184/SC4 1992). This kind of data model, which specifies the categories of information about an artifact during its complete life cycle, is commonly known as a Product Model (PM).

The goal of the Product Model is to provide a conceptual model that integrates physical product, activity, and resource information. The model focuses on two main abstractions: the relationship between products and activities, and the progression of an object from initial requirements to a proposed object, then to an actual realized object. Additionally, the Product Model works as the knowledge carrier to record all of the information associated with it during the whole construction period, even the entire life cycle of the product.

For simulation purposes, the Product Model should include:

1. Three dimensional topological information
2. Type and shape of the product component with its weight and center of gravity
3. Relationship between the product components
4. Construction process information

For information handling purposes, the product model must also provide a way to collect, store, and queue the information that will come from, and be used by, a variety of practitioners.

3.2.1 Product Atomic Component (PAC)

The Product Atomic Component (PAC) is defined as the smallest simulatable and constructable product unit. The significant amount of information available in CAD drawing allows construction engineers or construction managers to create Product Atomic Components (PACs) for simulating construction tasks (Fig. 3-2).

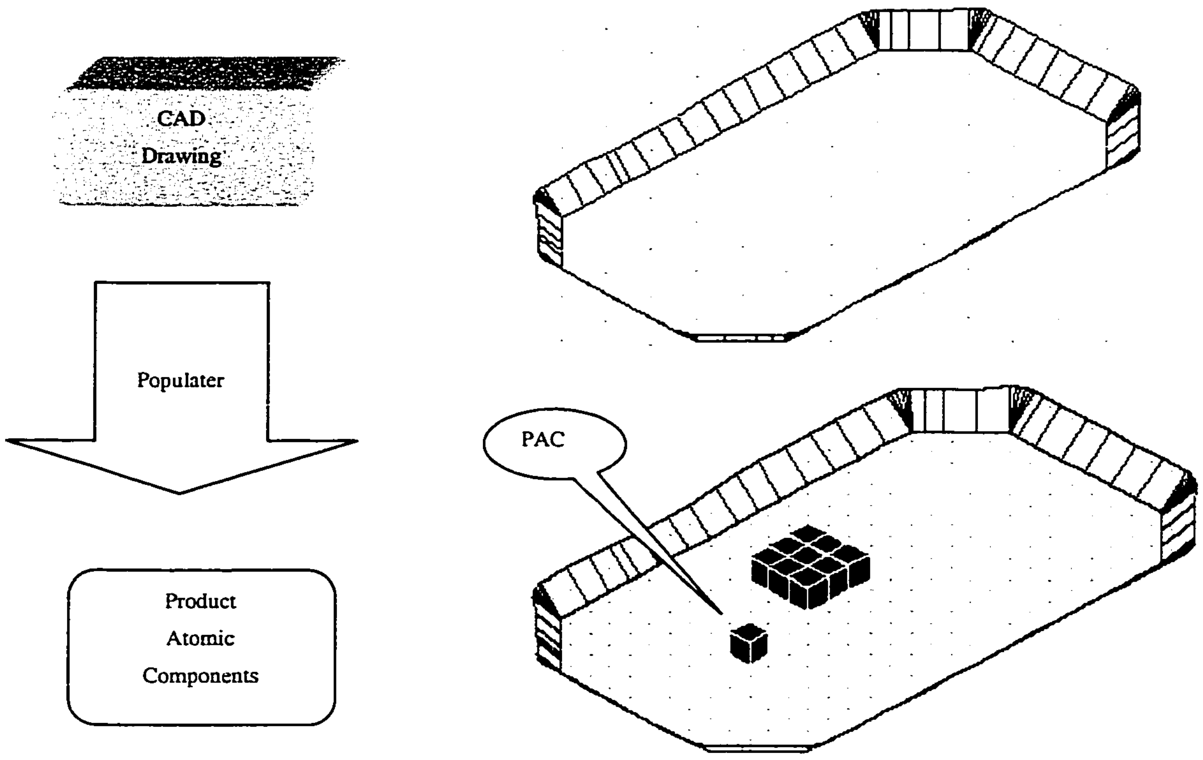


Fig. 3-2 Populator Model

The CAD-Populater is a plug-in for AutoCAD that assists with the development of the product model. The primary task of the CAD-Populater is to capture the 3D coordinate information stored in the CAD models and create PACs with sufficient information for simulation. The CAD-Populater model is responsible for capturing and generating data from three-dimensional CAD models and placing the interpreted data into the generated PACs.

Unlike most existing systems, which focus on transferring information – such as coordinate data and volume – from the CAD system to other applications, the CAD-Populater creates objects (PACs) within the CAD system, then transfers these objects to the simulation engine by taking advantage of ActiveX technology. These objects also act as the product-oriented ***Entities*** that share between the product-oriented simulation and process-oriented simulation.

Each generated PAC is stored in a database as a primitive solid object along with its associated attributes. These primitive objects can then be used to assemble a product hierarchy. Solid modeling representations give a complete and unambiguous definition of an object, describing not only the shape of the boundaries but also the object's interior and exterior regions. Further, solid geometry models are efficient in computational operations (e.g. calculations of volume and mass property). Table 3-1 lists CAD portion properties included in PAC.

Table 3-1 PAC Properties (CAD Portion)

Property	Description
ObjectID	Refers to AutoCAD Solid Object ID
CursorID	Used to indicate current PAC
Type	Object Type, Such as Sand, Clay, and Rock for soil type
Volume	Object volume
Quantity	Object Quantity
Remain	Object remaining Quantity
cx, cy, cz	Object geometric location
Length, Width, Height	Object dimensions

3.2.1.1 Identifying geometric objects within the CAD model

The most recent CAD systems are attempting to move toward an object-oriented framework, where the objects created in CAD drawings are elements of the project; i.e., all objects making up a facility, such as a dam or road. However, even though the object-oriented basis of representation is fully employed in the CAD systems, it still does not provide product atomic components that can be handled in a simulation model.

To support the Populater's creation of PACs, a front-end utility must be developed as a built-in tool for AutoCAD; this will allow users to identify geometric objects, such as a 3D surface model generated by Eagle Point, that hold geometric data used to create PACs.

3.2.1.2 Extract geometric information for each PAC

To get the geometric information, the populater module queries the CAD database for the selected objects. The information is used to create the PACs, to which more information will be added as long as the PACs proceed through the entire simulation process. The PACs created using this information will provide the CAD-ISE system with the fundamental components required within the object-oriented representation of the product.

3.2.1.3 Link CAD object to Product Atomic Component

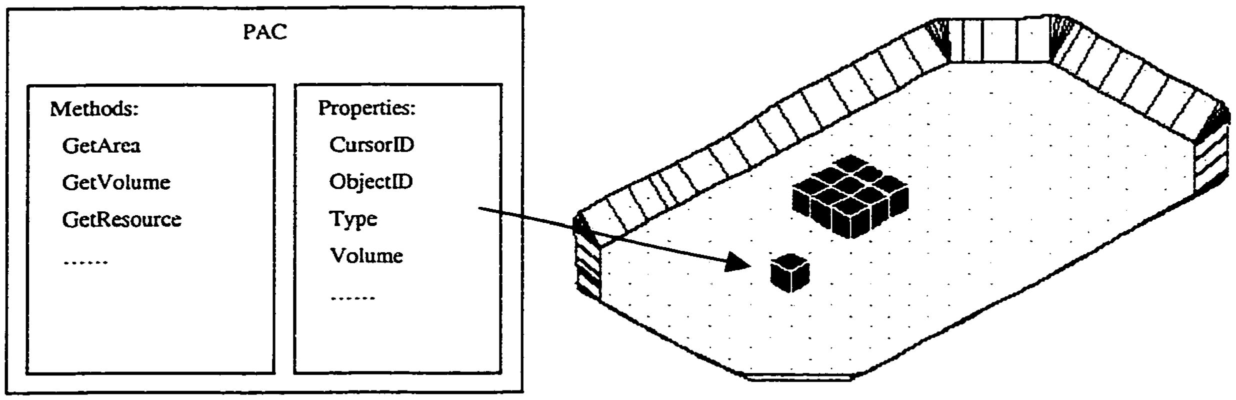


Fig. 3-3 PAC Object

The development of PACs to support the CAD-ISE system requires the capability of completely defining the geometric data from the CAD objects, while a product hierarchy structure deals with the relationships between the PACs. The geometric part of the PAC, including attributes, will be created in a CAD drawing and maintained by AutoCAD. The host part of the PAC will be created as an instance of the PAC class. The PAC class is in charge of the PAC's main properties, including a pointer linking geometric elements of the CAD drawing to the PAC, relationships between PACs, type, resource(s), construction method(s), etc., and behaviors such as calculating volume and area, generating estimate data, and creating schedules. The built-in pointer in the host section of the PAC takes care of the connection between the host and geometric PACs (Fig.3-3).

3.2.2 Product Hierarchy

The product representation schema performs the critical role of representing the information that is accessed through the relationship hierarchy, i.e., Product Hierarchy (PH).

Front-end utilities are required to provide users with an intermediate tool that creates logical groupings from the geometric PACs. Each PAC's position in the hierarchy is determined by the dependency that exists among components.

The Product Hierarchy (PH) matches the Product Breakdown Structure (PBS) that represents the construction product. In other words, PH is a representation of PBS in CAD-ISE. Taking earthmoving construction as an example (Fig. 3-4), the PH (i.e., the product itself) groups all working faces to represent the entire final product in a PBS. Working spaces represent excavation layers, i.e. Working Space 1 in the PH is Excavation Layer 1 in the PBS, and so on. Sections, which do not exist in the PH, are used to define the construction sequence in the Product Network (refer to the following section). For this example, the PAC is defined as a representation of the smallest construction unit with certain length and width that has the same soil type. Natural layer(s) under each excavation layer is (are) divided into certain PACs according to the dimension of PAC (i.e. Length and Width). One group of PACs represents one natural layer that belongs to a certain working layer. Together, all PACs represent the entire product, and detailed information will be stored in each PAC.

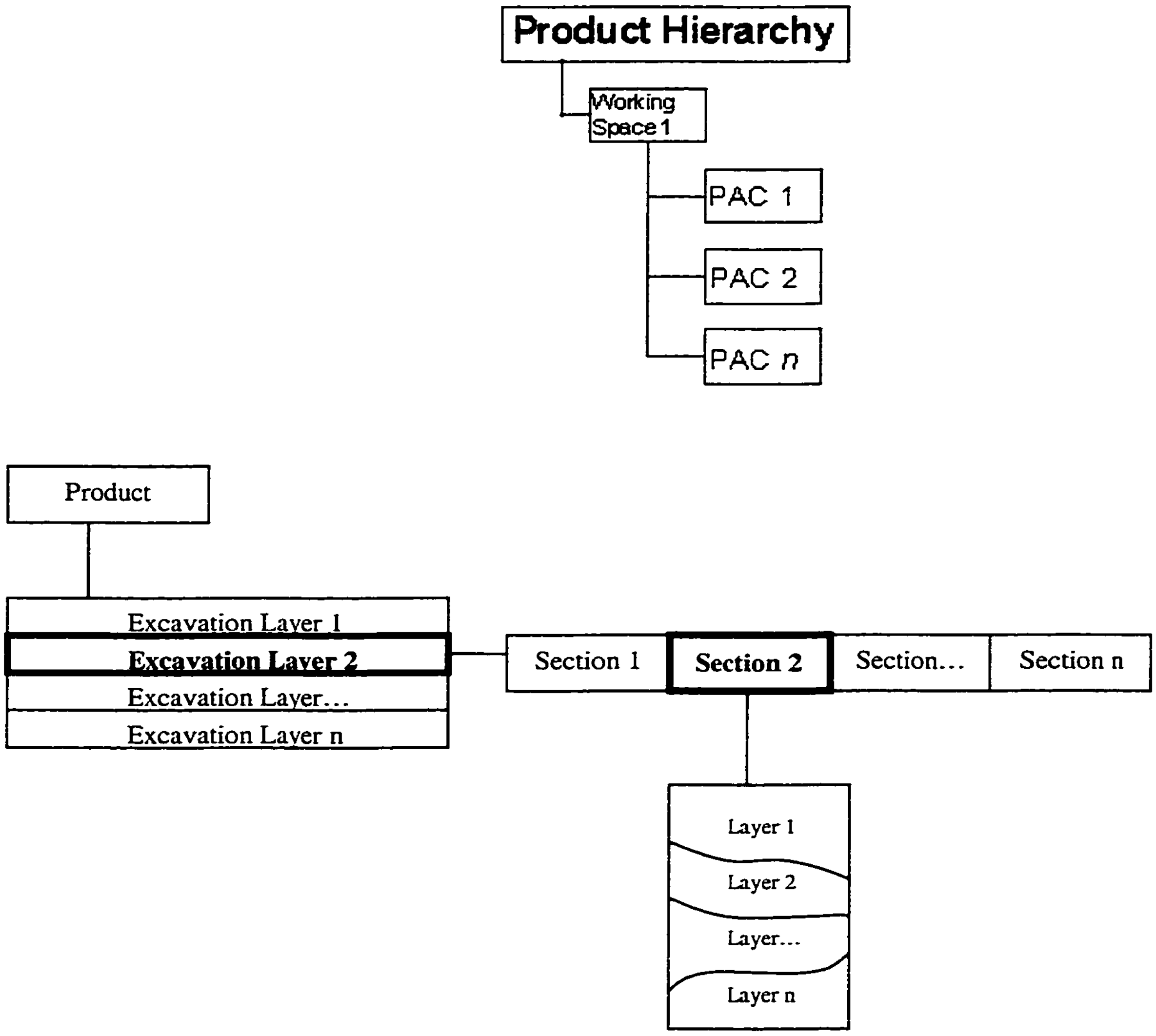


Fig. 3-4 PH vs. Product Breakdown Structure (Earthmoving)

3.2.3 Product Network

The intricate relationship between the different components of construction presents an incredible challenge for anyone that wants to simulate a construction project using a computer. The PH is not necessary to reflect the construction sequence of each PAC, as the sequence is determined by the order in which PACs are scheduled to appear in the environment. CAD-ISE will provide the utility to establish the Product Network (PN) based on the PH, which deals with the construction sequence. In construction operations, a process does not take place until the necessary prerequisites have been completed. The third layer of the excavation pit, for example, will not be removed until the second layer has already been excavated for the same section. The operations associated with objects in the PH, which take place at the same hierarchical level, will take place when the resources become available. The order in which these operations take place is determined by the priority associated with the object in question. If the objects have no priority, they are processed concurrently. After all operations within a level are completed, the next level of operations is initiated.

3.3 Resource Module

Resources are the driving force in construction operations. If time is recognized as a resource, then all construction methods constantly consume at least one resource. In simulation, simulation resources are defined as resources that are selected from the company's resource pool or rented from the outside and used or shared for the simulated project. The Resource Model (RM) acts as the bridge between the company resource management and simulation systems. RM should be able to access the company resource pool, manage the rental resources, and analyze the utilization of the resource for the simulated project. If the system runs out of required resources, the RM must send a message to the project manager. The resource hierarchy will thus be used to organize the resources. If the company resource management system can be extended or developed to include the above functionality, it will be an alternative to this model.

3.4 Construction Process Simulation Model

Simulation has been used successfully in a wide range of industrial applications. However, in construction applications, even though simulation has been applied, it is still extremely limited in terms of wide deployment in construction projects. Several factors limit the effective implementation of this technology in construction. Chief among them is the need of the construction engineer or project manager to be proficient at both simulation and the construction operation to be modeled. This also leads to the need for large amounts of time to be spent developing the simulation model. While developing models for industrial applications is just as time consuming, the perpetual nature of manufacturing activities, and the fact that there are often hundreds of thousands of the same product produced, make this investment worthwhile. However, this is not the case in construction.

To solve this obstacle to the application of simulation in construction, construction process simulation models play a significant role. These are a set of predefined models that represent typical construction processes and methods in terms of resources, including material, crew, and time. It provides the front-end user with a simulation template able to construct a practical simulation model by simply choosing and modifying these construction process simulation models. Excavating is an example of a common earthmoving process and is defined by specifying model, rate of excavator, productivity of excavator, or duration for a certain quantity.

Each PAC can have more than one process, and each process can be added in multiple ways, using either the process editor, which builds the process by using the atomic construction method and atomic simulation activity, or by simply selecting from the built-in construction process library. A process object can be simultaneously connected to any number of other PACs using any type of simulation construct, simply by invoking the methods (behaviors) of the object in various ways. In addition to this ability to use an object in multiple ways simultaneously, the new object automatically retains all inter-object connections whenever a new process is substituted for the current one.

The process embedded in the PAC can be used directly for simulation purposes. After conducting the simulation, the results, such as resources used and duration, become available and will be kept for future applications to achieve automation and avoid re-entry of data.

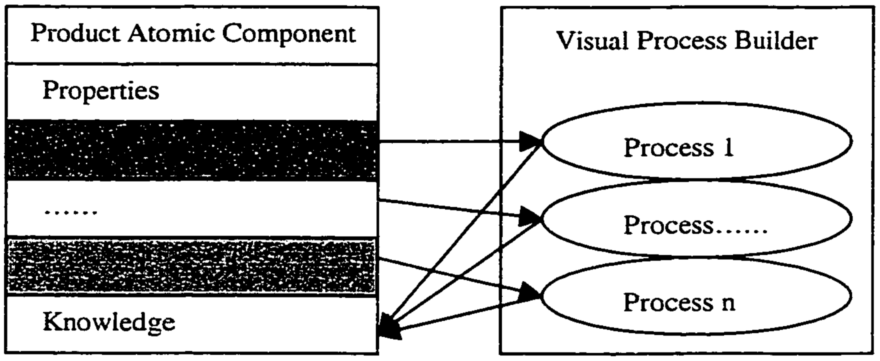


Fig. 3-5 Process in the PAC

3.5 Environment Model

The Environment Model (EM) is used to define site conditions and site layout, such as source, placement, and travel road. Geographical Information Systems (GIS) technology can be introduced for long distance travel, earthmoving/dam-fill, road construction, and paving.

3.6 Simulation Engine

CAD-ISE relies heavily on the underlying simulation engine. This special simulation engine is an event-scheduling simulation queuing model that arose as an object oriented implementation for special purpose simulation. It will provide run time support and repeatedly activate execution. The event-driven algorithm simply scans the module for an event that could possibly be scheduled for execution. The event selection is based on resource availability, logical constraints, other module constraints, and random phenomena. The selected event is routed to the appropriate process where future events can be scheduled. In addition to event scheduling support, it also provides numerous other forms of support, such as pseudo-random number generation (AbouRizk and Hajjar 1997).

3.7 Data Acquisition

CAD-ISE is needed to retrieve the process of the project and run simulation to revise the schedule at any place/time during the project construction period. The data acquisition module serves as the real data provider, collecting the daily site data and updating the PACs. Furthermore, the data acquisition module can be extended to collect more detailed information, such as labor cost, equipment utilization, and material consumed, for cost control purposes. The information obtained from data acquisition will also be used to re-calculate productivity. This new/updated productivity rate can then be used to conduct a more accurate experiment.

3.8 CAD-based Simulation Environment

The CAD-based Simulation Environment will provide the GUI (graphical user interface), with which the module user interacts to construct a model. The GUI links to: (1) the product model, providing all the product components and relationships between them; (2) the visual construction process builder, providing the means to create construction processes based on the standard processes and the built-in construction method atomic simulation models, which include the resources for each selected construction operation; and (3) the environment model, describing the site conditions, such as site layout and road. Through the environment model, users can experience different scenarios to aid them in decision-making.

3.9 Post-Processes

Since all information related to the product model, from its beginning until the final completion of construction, is collected in the product model, CAD-ISE can take full advantage of the data to provide many kinds of post processes, such as animation, scheduling, and estimating. This will help the user understand and analyze the simulation results.

3.9.1 Animation

Animation is an increasingly popular technique that can be used to understand, communicate, verify, debug, and validate simulation models. During the model development stage, animation is useful for verifying that the simulation model is behaving correctly and representing what the model developer had in mind. At the same time, it can be used to debug the model and discover why it does not behave as intended.

Visualizing the changing states of an entire system and tracing its behavior through a segment of simulation language code or a simulation trace output file can be extremely time consuming. In this respect, computer animation can serve as the ideal trace mechanism for revealing the complex inter-relationships represented by the model. Often the developer of a simulation model is not an expert in the process being represented. In this case, animation can be used for model validation, checking not only the model itself, but also the behavior of the system that the developer understands. Thus, animation can provide feedback and increase everyone's confidence, especially for those decision-makers who have no training and lack sufficient time to check the validity of models developed by others.

In addition, animation offers several minor benefits: (1) construction process visualization, providing a visual representation of the exact status of the project; (2) good communication, avoiding delays, error, accidents, etc.; and (3) constructability review, identifying design conflicts, analyzing scheduling conflicts, and evaluating construction alternatives.

Typically, the animation of a simulation model involves a graphical computer application that uses changes in the position, shape, or color of icon to illustrate changes in the state of the simulated system.

3.9.2 Schedule Generator

The schedule generator uses a model-based planning approach to generate different construction schedules based on the product network and simulation model. To generate the practical schedule, the system will present solutions to the following issues: (1) defining construction zones and groups of product components, (2) generating activities, (3) identifying construction methods for each activity, (4) assigning resources for each activity, and (5) sequencing activities.

The schedule generator is integrated into CAD-ISE to eliminate massive data input such as components, activities, relationships and cost information, and to directly access the design information in CAD. This integration overcomes the problem of massive data input without transforming design information.

Even though the schedule generator supports multiple levels of production components and activities, only the atomic components at the bottom level are considered. The group information and its relationships with components and activities are adapted to the different summary levels. This will reduce time and effort, and improve the accuracy of the schedule.

3.9.3 Estimate Module

The simulation system will not be utilized if it does not provide the modules to support estimating and scheduling functions. The CAD-ISE has all information needed to generate the estimate built directly into the model. The estimate module will be responsible for this, and will provide additional functionality that allows the user to produce customized estimates.

3.10 Conclusion

The presented CAD-ISE relies on the integration of CAD and simulation, through the application of object-oriented concepts, to provide an integrated environment that will allow practitioners to experience simulation without extensive knowledge of simulation or large amounts of data entry. Although the outline focuses on specific construction operations, the methodology and strategy developed here can also be applied to other applications. Furthermore, the classes developed here can be used to implement these applications with only minor modifications and the addition of more construction process simulation models to the library.

This outline presents the framework and methodology for CAD-ISE that will eventually be an entire CAD-based integration simulation environment. The system in its entirety will exceed the scope of one Ph.D. effort, so this work will focus on CAD-Populater, Product Hierarchy, Product Network, and the CAD-based simulation environment.

CHAPTER 4 PRODUCT-ORIENTED SIMULATION CONCEPTS

4.1 Introduction

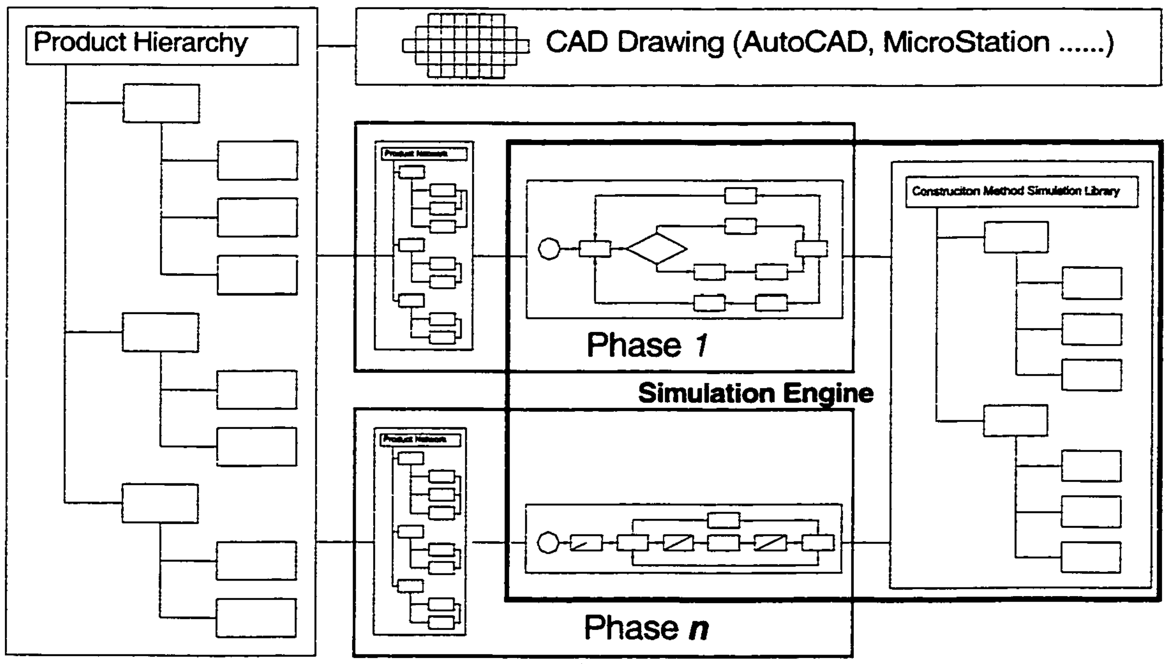
In the design of construction operations, it is often necessary to make decisions regarding complex construction processes. These decisions include determining crew sizes, selecting equipment, establishing operating logic, or selecting construction methods. Associated with each decision are a series of outcomes, such as construction cost and duration. Decisions are made based on their expected outcomes. For example, the equipment fleet to use in an earthmoving operation may be the one associated with the lowest expected cost.

Several techniques are available to assess the outcome associated with a particular method of performing a process. Experimentation with the real system, on one extreme, is very accurate but is expensive, slow, lacks generality, and is sometimes impossible to do. Mathematical modeling, on the other extreme, is very precise but requires that important aspects of the process be disregarded, requires a high degree of mathematical ability, and becomes too complex for most real life construction situations. Simulation is the third technique and is very convenient, because, while being realistic, it is also inexpensive, fast, and flexible.

4.2 Product-Oriented Simulation (POS)

This thesis attempts to investigate an alternative construction simulation methodology from a product-oriented perspective. This methodology focuses on the product, the simulated object, instead of the processes or activities. It specifically addresses construction needs by providing a flexible and open architecture that enables: 1) the detailed and efficient modeling of a complex system, 2) the efficient development of special purpose simulation tools for the construction industry, and 3) the simplifying of the simulation modeling process through integration. The CAD-based integrated simulation environment applies these concepts in order to provide a simulation environment that integrates with CAD, requires minimal simulation knowledge, and is simple to use. Utilizing CAD-ISE as a modeling environment, an engineer will be able to develop a model of the construction process for surface mine preparation, an earth-filled dam, and linear projects such as road and pipeline construction. The model can then be experimented with to study alternative construction methods, allocate resources, plan construction, estimate time and cost, and evaluate constructability issues.

4.2.1 Architecture



Product Hierarchy
CAD Drawing (AutoCAD, MicroStation ......)
Product Network
Construciton Method Simulation Library
Phase 1
Simulation Engine
Phase n

Fig. 4-1 Product-Oriented Simulation (POS) Architecture

To get a full understanding of the concepts of POS, many aspects must be addressed, such as product structure, construction sequence, construction method representation, and resources. In order to give a brief explanation and a clear global picture, the POS architecture is shown in Fig. 4-1.

The Product-Oriented Simulation Environment's applied methodology achieves integration between CAD, simulation, and other support systems in construction (e.g. estimating and scheduling) by adopting a product view of the constructed facility. In other words, the system representation and all of its abstractions are derived from the point of view of the product to be constructed.

At the essence of this approach are the product hierarchy, the product network, and the simulation model, which serve as the basis of all representations.

All relevant data related to the constructed facility is stored in the Product Hierarchy (PH). The PH represents the product's physical attributes, their relationships, and the methods by which they are constructed.

The Product Network (PN) is a concept used to define the construction sequence for each construction phase of the product, thus complementing the PH.

The Simulation Model (SM) of the project is created by linking the product model in a given construction phase to simulation models from a library of flexible or modular models. The SM also represents the methods by which the project will be constructed and the process to be followed. The relationship between the PN and the SM is created during this process.

Fig. 4-1 illustrates the POS architecture. The CAD program feeds information to the PH in an automated fashion, thus creating the product elements of the project. The

Product Network is built from the PH with specifications from the user to constrain how the product elements relate to each other from a construction point of view. An earthmoving example may be that the excavation of Section 1 must take place prior to that of Section 2. The construction methods library is then utilized to specify how the product will be built by referencing the proper process and tying it to the product network, thus creating the simulation model. The details of this interaction will be discussed in the following sections.

4.2.2 Product Representation – Product Hierarchy (PH)

The representation of the product is closely tied to the three most fundamental entities: the representation of construction products (e.g., a facility to be built and all of its physical components), the representation of the process used to construct them (i.e., construction activities), and the representation of resources used by the processes. A Product Hierarchy structure is used to represent the product (Fig. 4-2).

For a comprehensive representation of all physical elements within a project, the PH given in Fig. 4-2 is used. As demonstrated in the illustration, the project is broken down by first identifying the various "work spaces" forming the project (if required). A workspace is a physical description of a part of the project where construction will take place; it is constrained by the design, the underlying topological conditions of the site, and the technological limitations of the method of construction employed. The process can be automated utilizing parameters specified by the user, thus removing the burdensome task of identifying large numbers of workspaces from a project. The workspace is further defined by the PAC elements that make it up. Each of the PAC elements is then characterized by the various construction phases that are required to build them.

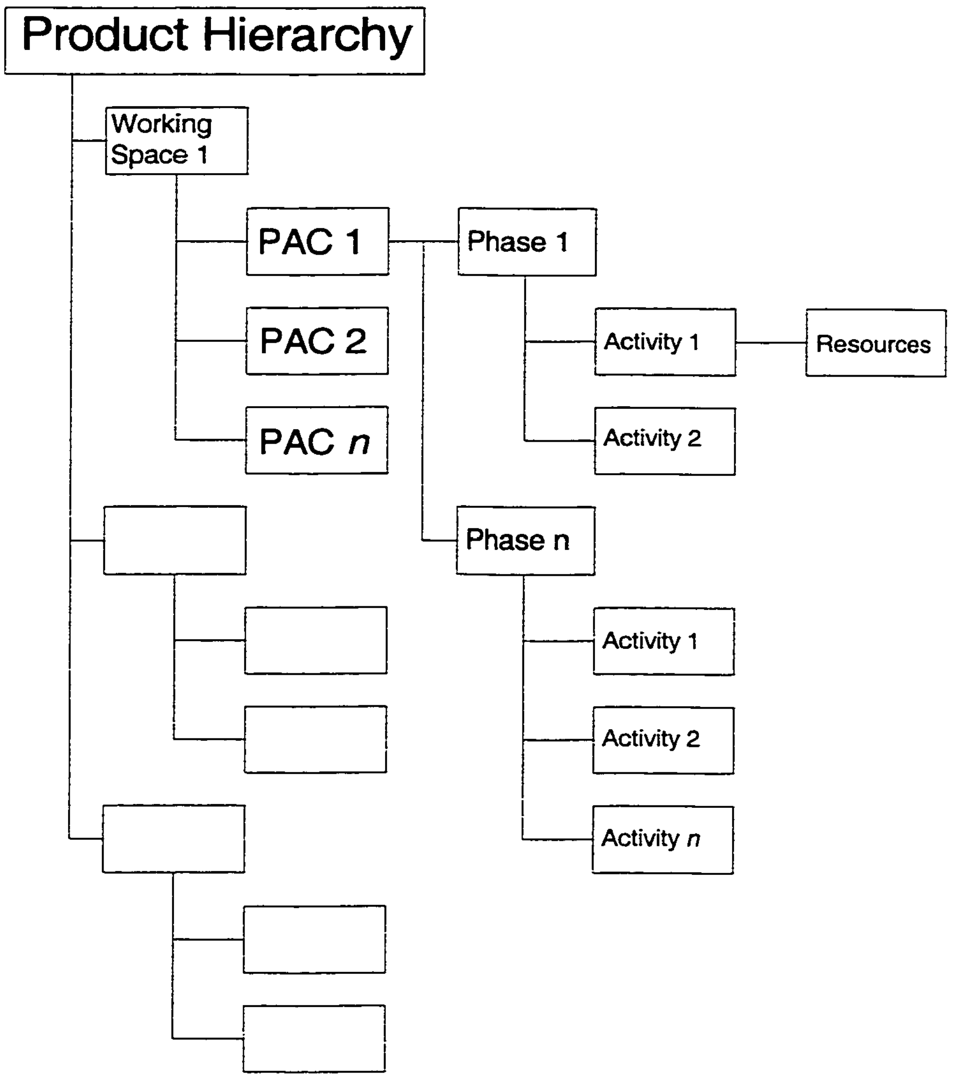


Fig. 4-2 Product Hierarchy Structure

The goal of the product hierarchy is to provide a model that integrates physical product attributes, process information, and resource information. The model focuses on two main abstractions: the relationship between the products and activities, and the progression of an object from initial requirements to a proposed object and to an actual realized object.

4.2.2.1 Product Atomic Component (PAC)

The object representation of the most basic element of the PH is the Product Atomic Component (PAC), which maintains and carries all information related to the product. To create the PH, each PAC that is generated from the CAD design drawing needs to be positioned in the hierarchy according to the dependency that exists among its components. Linking the product and the simulation model is achieved by building the relationship between them, as illustrated in Fig. 4-3.

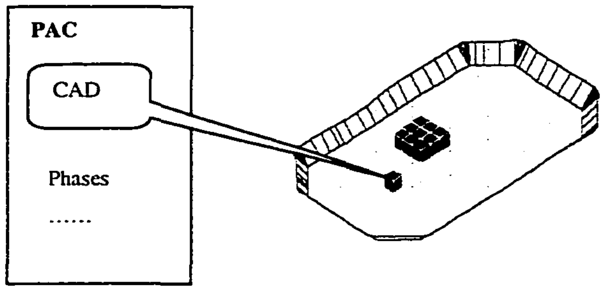


Fig. 4-3 CAD model in PAC

The CAD model resides within the native CAD environment, such as AutoCAD, where it was created. The model is extracted and embedded in the PH using a utility designed and implemented for this purpose. Information such as geometric data (position, attributes, etc.), volumetric data, and pertinent site characteristics are captured from the CAD model and stored in the PH for use within the simulation. Using the same approach, the data residing in the PH can be communicated to the CAD model during or after simulation, for the purpose of model update or presentation.

4.2.2.2 Construction Phase

Each PAC may pass through a number of different construction phases or operations. A construction phase represents a period of the life cycle of the PAC where it is transformed into another product. The project phase is defined by the activities required to complete the phase. Activities are completed with resources, as shown in Fig. 4-2.

Taking a typical earthmoving project as an example, three construction phases are usually involved: 1) preparation; 2) moving, which includes excavating, hauling, dumping, and returning activities; and 3) spreading, which includes spreading and compacting activities. In each phase, the construction sequence could be completely different. For example, the preparation could be done in parallel, the moving could be done radially and/or vertically, and the spreading could be done in any direction within the same layer. Therefore, these construction phases cannot share one construction sequence definition. In order to address this issue, a collection of phases is used to represent the construction phases that define the individual construction sequence through which each PAC will process (Fig. 4-2).

4.2.2.3 Construction Activity

In the product-oriented simulation approach, the construction process is defined as the activity that acts upon the PACs. Each PAC will proceed through several activities within every separate construction phase (Fig. 4-2). The activity collection object in the phase object is used to collect the activities that the PAC will process through during this phase.

Taking earthmoving as an example, PAC will be certain amount of dirt, and the activities that process through it would be preparing, excavating, hauling, dumping, compacting, or part of them.

Sometimes, several parallel activities represent one occurred activity in terms of PAC. It is common to have two excavation activities; for example, one deals with rock using a power shovel, and another deals with clay using an excavator. The actually occurring activity is subject to the PAC type (e.g. soil type). In other words, it is decided during the simulation run time.

4.2.2.4 Resources

Resources are the driving force in construction operations; if time is recognized as a resource, then all construction activities always consume at least one resource. The resources object in the activity collects the resources required by the activity, and resource objects are always the end leaf of the Product Hierarchy structure.

The PAC hierarchy structure defined above accumulates all necessary information associated to it, giving:

1. The physical PAC, which carries out the processes;
2. Productivity and unit cost rates for the activities;
3. Locations where the activity will occur (CAD object in PAC carries detailed geometric information regarding the PAC);
4. The resources used by the activities;
5. The relationship between PAC, activities, and resources.

This aggregate hierarchy product model, which collects the PACs together, properly represents the whole production and acts as the information repository.

4.2.3 Construction Sequence Representation

Construction sequence can be categorized in two ways: as a product-based construction sequence, which deals with the constructed order of PACs, or as a process-based sequence, which deals with the sequence of the activities acting on the PACs.

4.2.3.1 Product-based Sequence

In construction operations, a process does not take place on a given component until the necessary prerequisite components have been completed. The intricate relationship between the different components of construction presents an incredible challenge to anyone who wants to simulate a construction project using a computer. The PH represents the product's physical relationships, covers all of the construction phases of the product, but does not reflect the construction sequence of each PAC. Practically, it is impossible to use the product hierarchical structure to define the construction sequence for each construction phase.

The product-based construction sequence is defined as the order in which PACs are scheduled to be constructed, based on each construction phase. To address this issue, a Product Network (PN) is introduced.

The PN outlines the logical relationships between the physical components of the project, offering a visual means by which the order of construction for the PAC elements is specified. A PN is generally required for each phase of construction as the specified product is evolving. Once complete, the PN contains the sequence of all PAC elements for a given phase of construction.

To facilitate better project definition, the concept of a "construction section" is introduced. A section defines a group of PACs that can be constructed together and are only constrained by other sections. A space buffer between sections can provide further constraints to ensure that a whole section is complete and a certain time has elapsed prior to the start of another section.

The predecessors and the successors defined in the PN could be working-spaces, sections or even PACs. According to the definition of predecessors and successors in the PN, components in the same hierarchical level will commence when their predecessors have been complete and resources become available, according to the order defined in the PN. If there are no predecessors assigned, the component will be undertaken based on the order of PAC creation in the PH. The construction sequence for each component can be automatically determined at creation time for simple projects. After all components within the predecessor have been completed, operations in the successor are initiated.

To illustrate the concept of this product-based construction sequence, a simplified PN for a project is shown in Fig. 4-4. There are three working spaces defined at Level 1, i.e., Working Space –1, Working Space – 2, and Working Space – 3; three sections at Level 2, i.e., Section – 1, Section – 2, and Section –3 and various PACs (PAC-1, 2, 3, 4, 5, 6, 7) included in each section at Level 3. Since the components at Level 1 (working space) share a common parent, they have the same priority, and their construction sequence is defined by their predecessors and successors. According to Fig. 4-4, there are no predecessors or successors defined; therefore, the sequence will be as is (Working Space-1, Working Space-2, then Working Space-3). In the case of Level 2, Section-3 is the predecessor of Section-1, and as a result, even though Working Space-1 is ahead of Working Space-3, Section-3 will be carried out before Section-1. As illustrated here, the lower level sequence governs the higher-level sequence.

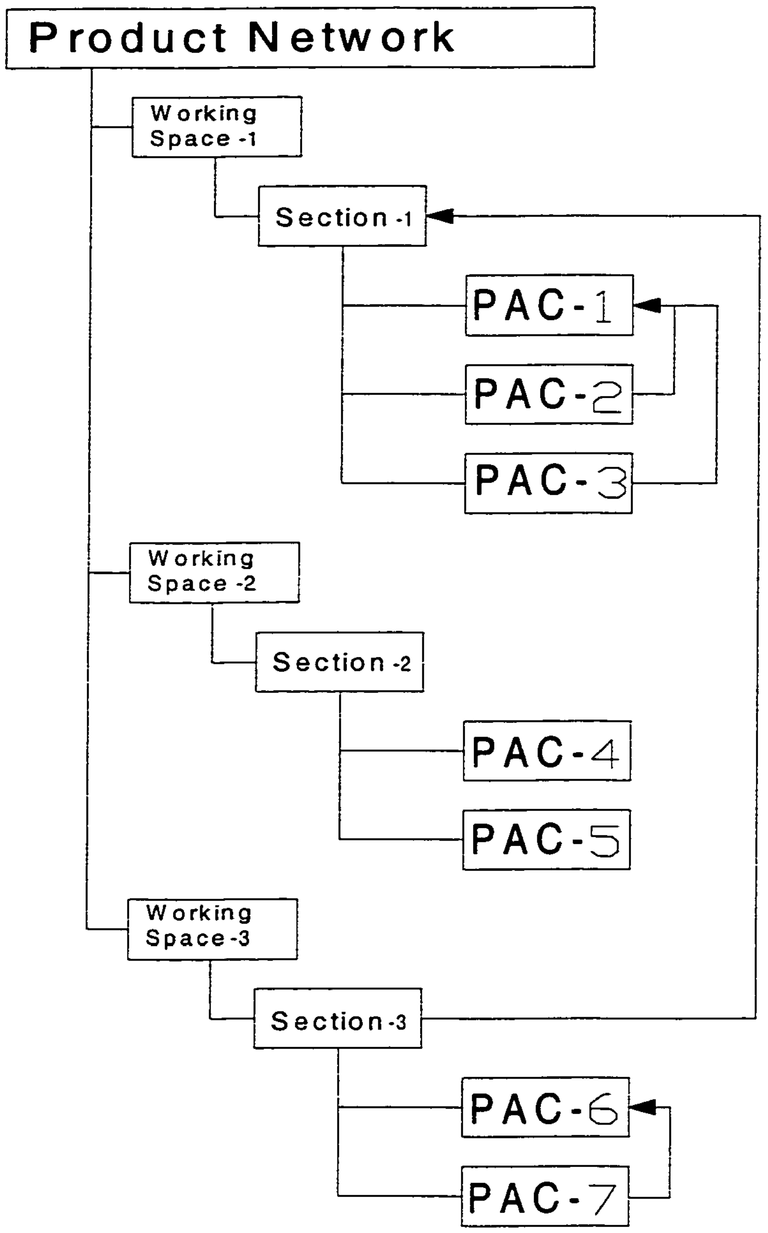


Fig. 4-4 Product Network

The lowest level is the PAC level, which eventually determines the construction sequence of each PAC. The sequence is determined by:

1. Predecessors and successors defined in this level. For example, in Section-1, there are three PACs: PAC-1, PAC-2, and PAC-3. PAC-2 and PAC-3 are the

predecessors of PAC-1 (or, PAC-1 is the successor of PAC-2 and PAC-3). Therefore, PAC-2 and PAC-3 are scheduled before PAC-1.

2. Predecessors and successors defined in higher levels. Because Section-3 is a predecessor of Section-1, PAC-6 and PAC-7, the children of Section-3, are scheduled ahead of PAC-1, PAC-2, and PAC-3, the children of Section-1.
3. The order occurring in the PN. For instance, PAC-4 and PAC-5 have no explicitly defined sequence, thus, the sequence will be that of the PN (i.e., PAC-4 ahead of PAC-5).

In short, the final construction sequence in Fig. 3-4 is:

PAC-7 → PAC-6 → PAC-2 → PAC-3 → PAC-1 → PAC-4 → PAC-5.

4.2.3.2 Process-based Sequence

The process-based construction sequence is defined as the sequence in which activities operate on a PAC within one construction phase. The activities in different construction phases do not have a forced sequence; in other words, there are no defined sequences for activities that belong to different construction phases.

Phase 1	Preparation					
Phase 2		Excavating	Hauling	Dumping	Returning	
Phase 3					Spreading	Compacting

Fig. 4-5 Activity Sequence among Construction Phases

However, if the activity is declared as the governing activity (defined in the activity modeling element), this activity must finish before the next construction phase can start on the same PAC, section, or working space, depending on the PN definition.

An earthmoving operation, for example, includes three phases – preparation, moving, and spreading – and each phase has its own activities (Fig. 4-5). In Phase 1, preparation is the only activity and is a governing activity, which means the following phases only can begin after this activity has finished. In Phase 2, there are four activities: excavating, hauling, dumping, and returning, where hauling and dumping are governing activities and returning is a non-governing activity. This means that the third phase can start after the dumping activity is finished, without regard to whether or not the returning activity is finished. Furthermore, the governing activities decide whether the individual PAC is completed or not.

4.2.4 Simulation Modeling

A simulation modeling toolkit is needed to support the requirements identified in the product-based simulation methodology. This simulation modeling toolkit is required to provide not only run-time simulation support but also an open environment that will allow a developer to create a suite of required modeling elements based on the product-oriented simulation methodology. The special purpose simulation approach is adopted for simulation model representation, which has the advantages of focus and specialization for integration purposes, and simplicity, from a user's perspective.

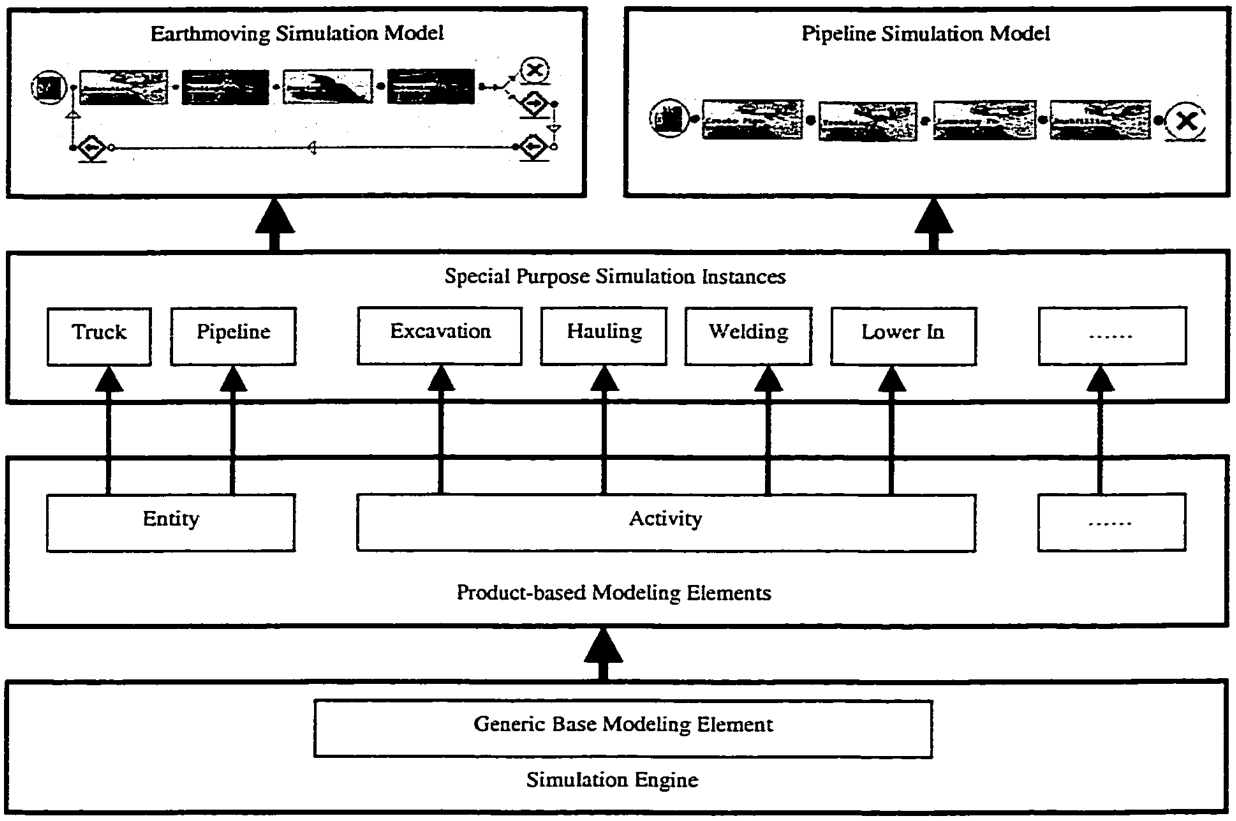
4.2.4.1 Engine

The strategy of implementation is based on Simphony (Hajjar and AbouRizk 1999), which is a discrete event simulation system that arose as an object-oriented implementation for special purpose simulation. The event-driven algorithm simply scans the module for an event that could possibly be scheduled for execution. The event selection is based on resource availability, logical constraints, other module constraints, and random phenomena. The selected event is routed to the appropriate process where future events can be scheduled (AbouRizk and Hajjar 1997).

In addition to event scheduling, Simphony provides numerous other forms of support, such as a "low-level" modeling element designer that allows a developer to develop special modeling elements using Visual Basic scripting (VB-scripting), and a simulation model editor that allows users who do not have a simulation background to build the simulation model, based on built-in modeling elements.

4.2.4.2 Product-Oriented Modeling Elements

The main challenge in construction simulation is providing tools that are easy-to-use for inexperienced users, and powerful enough to model the details of construction processes. As such, the focus of the simulation model for this research will be to simplify the modeling process by providing a variety of less and more powerful product-oriented simulation components.



Earthmoving Simulation Model
Pipeline Simulation Model
Special Purpose Simulation Instances
Truck
Pipeline
Excavation
Hauling
Welding
Lower In
......
Entity
Activity
......
Product-based Modeling Elements
Generic Base Modeling Element
Simulation Engine

Fig. 4-6 Relationship between Engine and Modeling Elements

Modeling elements are made to resemble the corresponding physical or logical real world elements from which the simulation model will be built. In other words, a simulation model becomes a collection of instances of modeling elements. Modeling element instances based on the same modeling element would share the same code and are identified solely by the value of their properties, which are specified by the user as they are created. Fig. 4-6 illustrates this definition and the relationship between the engine and the modeling elements.

4.2.4.2.1 Entity

Generally, an entity is defined as an object moving through the simulation model. An entity could represent a part, person, or message. It can also be thought of as an information packet with a set of attributes.

Entities are introduced into the process through the Entity modeling element, which specifies how many and how often entities arrive at the process. Once an entity has entered the process, its future course of action is determined by the processing instructions contained in the activities and routings; typical instructions include processing delays and move times. Entities that have completed all processing instructions are expunged from the system, and its statistics are collected.

A Product-based Entity (PE) is an object that represents a piece of the product. A PE could be one PAC defined in the Product Hierarchy, part of a PAC, or even several PACs, depending on the capacity of the entity defined in the simulation model. Therefore, the construction simulation is nothing more than the PEs flowing through the activities that model the construction processes.

Beyond the common functionality that supports the normal activity-based simulation, in the product-oriented simulation context, the entity used in the engine should be enhanced in order to carry PEs along with it. As the entity flows through the simulation, the PE would also proceed through each construction process in the model. After the PE experiences all of the activities that should be undergone in the construction phase, the construction phase is completed.

If the focus is on the simulation engine, the entity can still form a cycle as in the activity-based simulation model. However, if the focus is on the entire project, the entity (i.e. PE) is always the product and not the cycle, hence the term product-oriented simulation. Fig. 4-7 demonstrates the relationship between the Product-based Entity (PE) and the regular entity.

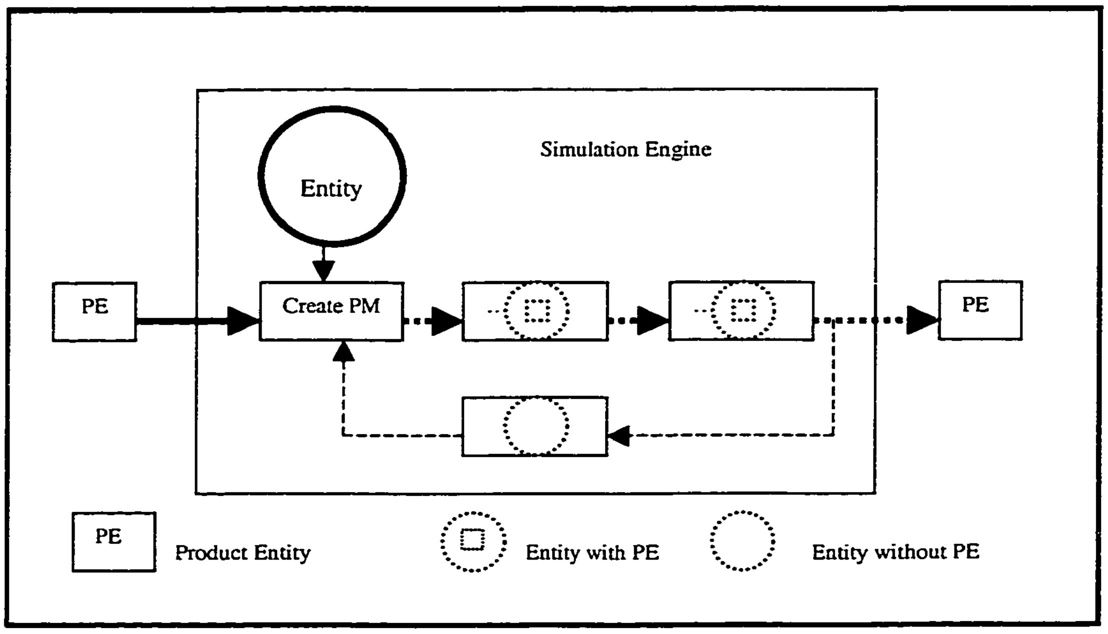


Fig. 4-7 Product-based Entity and Regular Entity

The entity in the simulation engine is created by the ENTITY modeling element, and as such, is an instance of the ENTITY modeling element. It could be anything, even a dummy element; it simply serves as a container to carry the PE, which is created outside of the simulation engine, to perform the simulation. The entity created in the simulation engine is not the controller; instead, the PE carried by the entity controls the whole simulation process. However, the information used to create a PE, such as measure, site created from, and quantity, is entered into the system by specifying the attributes of the entity modeling element.

4.2.4.2.2 Activity

Activity is the modeling element that represents work or a task to be performed using necessary resources. Each instance of an activity has its own duration that represents how long it takes to do the associated work. Activity instances also hold those specific resources that were acquired in order to start it. All construction methods can be built by specifying the parameters such as productivity and duration, which define the activity's character, from the common activity-modeling element.

Products are transformed from one status to another via activities, which may or may not correspond to physical locations in the system. Associated with each activity is the logic necessary to describe the actions required to process each entity through each activity in the process. This logic may be as simple as a time delay (only consuming time) or as complex as a nested if-then logic statement that chooses one processing time over another, based on the attribute values of the entity within the process.

A Product-based activity works not only as regular activity, but must also provide the following special features that present the synthesis of the product and simulation model:

1. Creating the Product Model (PM). The product-based activity in POS has an option to choose whether it needs to create the PM or not before the activity is started. The creation function in the activity triggers the method, implemented in the product class, to create the PM, based on the information carried by the entity, if there are enough products available. Otherwise, the entity will be added to the queue, and wait until the prerequisite activities or phases provide more products, at which point will be fired again. While creating the PM, the create method checks both the PH and PN to get the product and construction sequence information respectively. This feature allows the product model to

control the entire simulation process. The create event in the activity modeling element will be responsible for this functionality (Table 4-1).

2. Allocating and releasing the resources. The activity modeling element used in the POS integrates the allocating and releasing of the resources to provide an ease-to-use environment. This environment does not need a user to allocate and release the resources separately by using the additional modeling elements, such as capture, request, await, or free. It also offers a straightforward and natural way to assign resources to the activity by adding all required resources to the activity as children (Table 4-1).

3. Communicating with the PN and PH. The product-based activity supports complex messaging mechanisms that explicitly send information to the PN and PH. This information reflects the on-going state of the simulation, and in POS, these messages are hidden from the modeler. From the modeler's perspective, this suppresses unnecessary complexity. As the entity transfers into the activity, the activity will send a message to the PN (and through the PN to the PH) to inform the PN and PH that the activity is starting. Before the entity transfers out, it also sends a message to the PN and PH to inform them that this activity is complete. In the meantime, information such as start time, end time, and resource usage is collected and stored in the PH for the later use. All these functions are achieved through Activity Events (Table 4-1), which invoke the methods in the product-based entity (ActiveX Object).

Table 4-1 Activity Events

Name	Function	Schedule
Create	Create the Product Model by checking PH	Allocate
Allocate	Allocate the required resources to start activity	Start
Start	Start activity and update the status of PH	Finish
Finish	Finish activity and update the status of PH	Release
Release	Release the hold resources	Allocate

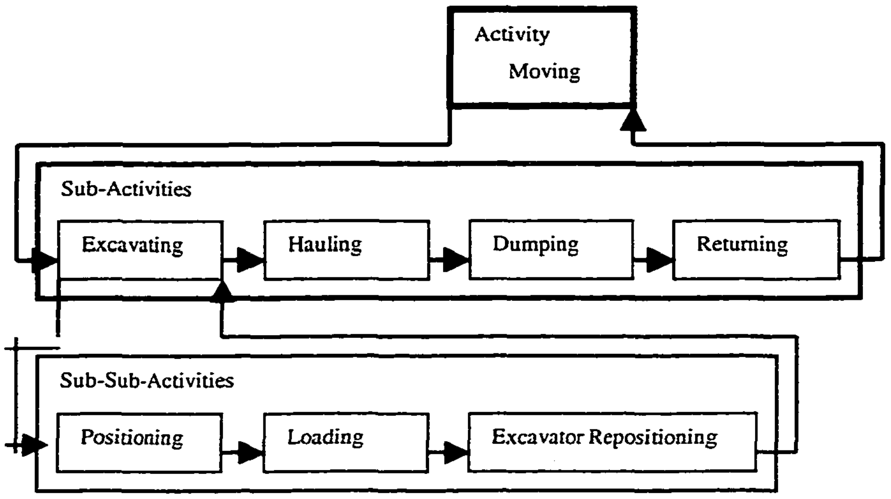


Fig. 4-8 Activity Hierarchy Structure (Earthmoving Example)

4. Building the construction method hierarchy. The hierarchy structure of the activity allows for the flexibility of running simulation at a different level. With the hierarchy feature, each activity can have its own sub-model that is made up of other low-level activities, and the low-level activities can drill down further until the desired level is met. This feature provides a means to design the construction method hierarchy in such a way as to significantly simplify the modeling process (Fig. 4-8). The decision as to which level it should go depends upon how much information is available and how much detail is desired.

4.2.4.2.3 Resource

Resources are agents (typically people or equipment) that are used to process products at activities and/or move products from one activity to another. Activities and entities may require multiple resources, or may choose amongst several available resources, based on the occurrence of a user-defined condition.

There are two kinds of modeling elements used to specify the resources: 1) the Declaration Modeling Element, used to register the resources to the simulation model, and 2) the Request Modeling Element, used to make the request for resources for each individual activity or entity. The scope of usage for the resource depends on where it is declared: the level where the resources are declared and the next level down can access the resources.

Resources can be requested by the activity, specifying the resources required by the entity to perform the activity; eventually, the resources specified in the entity will be added to those activities that the entity passes through. In other words, the entity does not consume the resources, it simply carries the information that identifies itself as a kind of resource that must be recognized while it passes through each activity (i.e., be added into the activity's resource collection).

The resources requested in the activity are allocated before the activity starts, and they are released after the activity has finished. If the resource is a type of material, it will not be released, because it becomes part of the product. The rules that guide the resource allocation are implemented by specifying the parameters – such as priority – in the Request Modeling element.

4.2.4.2.4 Routings

Routings specify how entities travel from one activity to another. Common routing types include percentage and conditional. Percentage routing allows the user to specify the probability of the entity processing the specific activity. Conditional routing is a kind of decision point, where conditions are set. If certain conditions are met, the following activity will be processed. The routings offer a solution to control the entity flow and provide a flexibility used to match real world situations.

4.2.5 Construction Model Simulation Library (CMSL)

In Complex modeling processes, which require proficiency at both simulation language and the construction operation being modeled, the large amount of time needed is the chief obstacle hindering the application of simulation to the construction industry. In order to solve this obstacle, the template methodology is implemented, using the construction model simulation library. This method is expected to offer an easy way for the front-end user to create the simulation model through picking and modifying the standard construction simulation model.

POS furnishes a Construction Model Simulation Library (CMSL), which is made up of common simulation modeling elements, such as entities and activities, as well as special purpose simulation modeling elements derived from common modeling elements (e.g. Excavating from activity element and Truck from entity element). Most of these elements have attributes that users can manipulate to modify the outcome of the simulation. Each modeling element provides a specific functionality and allows users to manipulate its internal behavior through its parameter attributes.

POS is intended to simplify the modeling process and automatically generate models. To achieve this, the specific construction method simulation model library that targets a specific type of project, such as dam, earthmoving, and pipeline installation, is built into the system. The user then develops the simulation model by selecting construction methods from a simulation library and assembling them into a final product-oriented simulation model. Models are built hierarchically in a top-down fashion. Higher-level elements generally represent overall operations, while lower levels incorporate more details regarding the specific processes. Fig. 4-9 illustrates the CMSL structure.

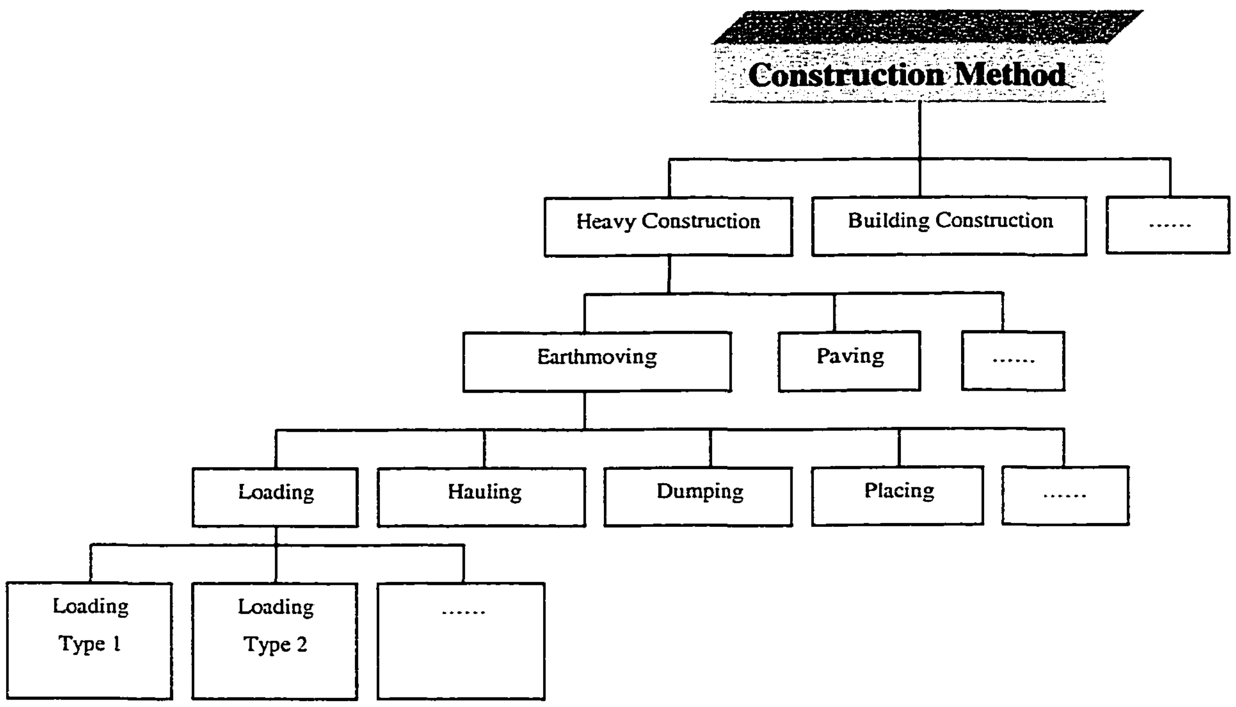


Fig. 4-9 CMSL Structure

4.2.6 Process Description

Running product-oriented simulation is similar to running activity-oriented simulation, with the addition of message flow and simulation control, which deal with the communication between the Product model and simulation model, and controls the entire simulation process respectively.

4.2.6.1 Message Flow

In Product-oriented Simulation (POS), the message not only flows inside the simulation engine, but also flows outside the simulation engine. In other words, messages should be sent in and sent out of the simulation engine to establish communication between product model and simulation model. ActiveX technology is an excellent solution, as it allows one object to be passed into another object. Here, the product-oriented entity is an ActiveX element that is passed into the simulation engine and attached to its entity. While the entity proceeds through each activity, the events in the activity will be fired, based on the schedule sequence, and a message will be sent through, calling the method in the object.

First, the create event is fired and a create PM message is sent out. Once the PH receives the create message, the create method will be executed and the result will be sent into the activity instance, which sends the message out. Whether or not the next event is scheduled depends on whether or not the PM is created. If the PM is created successfully, the allocate event will be scheduled without any delay. After the activity is allocated its required resources, the start event is fired. The start message will then be sent to the PH through the PN without feedback. Finally, the finish event will send a finish message to the PH and update the status of the product while the finish event is fired. Fig. 4-10 graphically represents the message flow.

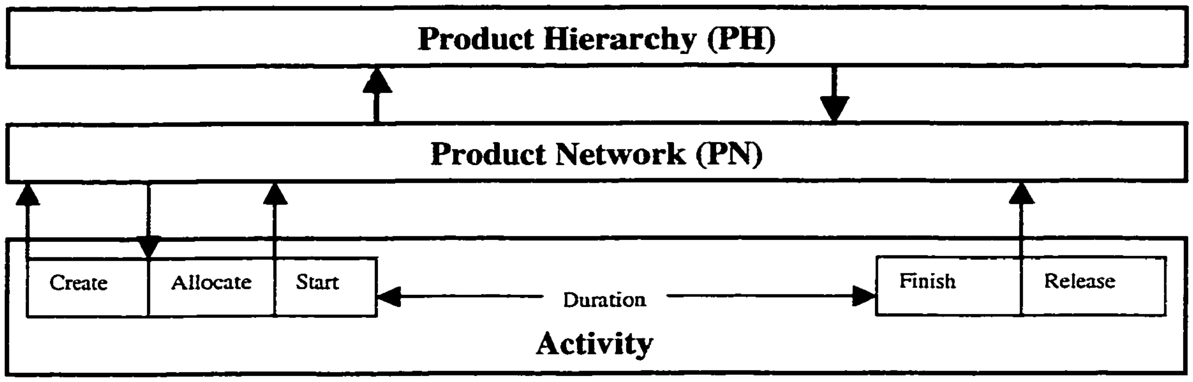


Fig. 4-10 Message Flow

4.2.6.2 Simulation Control

In the Product-oriented Simulation (POS) context, both Product model and simulation engine can control the simulation. Simulation can be terminated by the user through the simulation engine or when the model has run out of product. However, if the simulation is terminated by the user, the simulation does not finish completely.

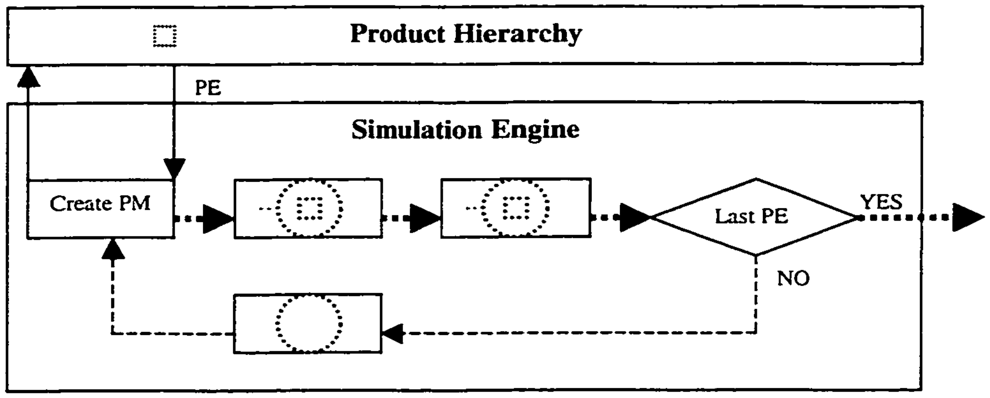


Fig. 4-11 Simulation Control

Usually, the entity will be created or awakened once simulation starts. The entity can begin its cycle and continue carrying PEs through the simulation model. After carrying the last PE through the simulation model, the entity will be destroyed. If there are no active entities in the system, the simulation will terminate (Fig. 4-7, Fig. 4-11).

4.2.7 Benefits of POS

POS methodology aims at addressing the issues that affect the application of simulation in construction industry, and attempts to offer some kind of solution to these issues. The benefits derived from this methodology will enhance the application of simulation within the construction industry.

POS makes full integration of CAD and simulation possible, which results in data sharing and lifecycle data collecting. CAD-ISE developed based on POS provides pre-defined common construction simulation templates and atomic construction method models that mitigate the increased complexity of simulation modeling.

4.2.7.1 Engineer's Perspective

POS offers a solution oriented to the engineering process. The simulation entity is always the product itself with which engineers are familiar. All pre-defined modeling elements used to assemble construction the simulation model are construction activities, such as preparing, excavating, and hauling without simulation-oriented elements, such as wait and free. The simulation model is built based on construction concepts from the engineer's point view so they do not need to think in terms of simulation concepts while they are building the simulation model. They can focus on the way in which they are constructing the facility. This will importantly free the engineers from unnecessary training in simulation concepts.

4.2.7.2 Intelligent information manipulation

A product-based entity serves as a depository of information that can be manipulated intelligently, while building the model of the actual product in such a way as to query the entire project. The user can thus query any individual product to review its properties, or simultaneously query all designated products for any property value. The properties also cover all engineering concerns, including start time, finish time, duration, and resources.

For example, for an earthmoving simulation, the simulation results stored in the PACs include duration and usage of excavator, trucks that work on the corresponding PAC, start time, and finish time. The information collected in PACs will be used to generate estimates and scheduling later on.

4.2.7.3 Lifecycle information Collection

POS fully integrates applications associated with the project and provides a means of collecting lifecycle information for the facility from design to maintenance since the product itself is capable of data storage. Therefore, all information regarding the product can be collected, stored, and queried.

Utilization of information from CAD design drawings reduces data preparation and entry tremendously. Based on this lifecycle information, the simulation can start from any point after the project start time and take advantage of the information collected from the project. This will improve the simulation accuracy significantly and guide construction more efficiently.

4.3 Conclusion

Product-oriented simulation methodology provides an alternative way to simulate the real world, as it looks at the whole picture, rather than just at the construction process itself. This will significantly impact the way in which simulation will operate.

This chapter conceptualizes the product-oriented simulation methodology in detail. It provides the architecture, defines terms, discusses major components of POS. The following chapter will describe the implementation of the prototype system developed based on the concepts documented in this chapter.

CHAPTER 5 PROTOTYPE SYSTEM IMPLEMENTATION

5.1 Introduction

To prove the product-oriented simulation methodology and the integration of simulation and 3D-CAD, a prototype system is developed. The primary aim of the prototype development and case applications was to prove the feasibility of developing CAD-based integrated simulation models adopting product-oriented simulation concepts.

The implementation of a CAD-based Integrated Simulation Environment is categorized into five distinct components, as illustrated in Fig. 5-1 and described in Chapters 3 and 4.

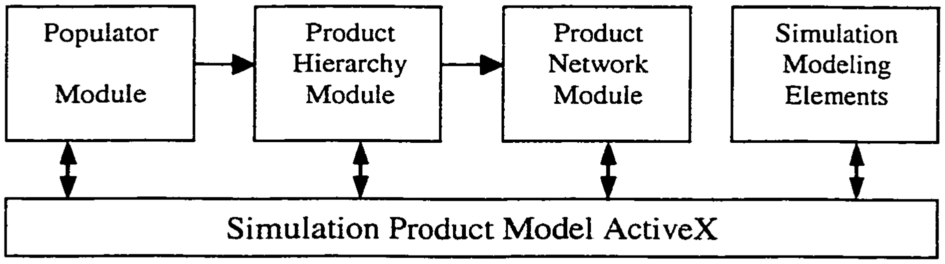


Fig. 5-1 Implementation of CAD-ISE

This section presents the implementation of a prototype object-oriented system for integrating a 3D-CAD and simulation model. According to the research objectives and the problems addressed by this research, the functionality and capabilities to be provided by the prototype CAD-ISE are first recognized. Section 5.2 briefly describes the system architecture and functionality. The CAD-ISE implementation that includes identification of system components, definition of the system module, and detail specifications of the main functionality of the prototype CAD-ISE is presented in Section 5.3.

5.2 Prototype System Architecture

The internal design of CAD-ISE is such that there are five distinct modules developed to accomplish different tasks, as illustrated in Fig. 5-1.

Populater Module provides a tool that runs under the CAD environment for the development of the Product Atomic Component (PAC). It offers the capability of constructing and re-constructing the PAC by using the tools built into the CAD system, such as working space create, natural surface create, grid generate, and linear product model generate. The integration between CAD and simulation is achieved by sharing the same PAC within both systems. In other words, the populater is a kind of bridge program that establishes the relationship between these applications.

Product Hierarchy Module is employed to build the product hierarchy structure used to integrate information from the 3D-CAD drawing and the simulation feedback during the course of a simulation run. Furthermore, it provides a means to define construction methods (Activities) for each PAC, and the associated resources that are used to operate them, by accessing and extracting construction method information from the corresponding simulation model. The module is also equipped with the capability to serve as the information carrier, which stores all information with respect to the PAC – from the initial stages of design to the completion of construction – such as physical geometry, location, activities, resources, start time, finish time, and duration. The module is in charge of updating PAC status, used to determine whether a specific PAC can start, is starting, or has already finished, by assessing the information collected in the PH.

The Product Network Module allows users to visually define the construction sequence among all PACs. It creates a friendly environment combined with the CAD system, which lets users define the predecessor(s) and successor(s) of each PAC in an interactive and visual way (here, predecessor and successor refer to the PAC rather than the activities). It is responsible for handling the relationship among all PACs included in the project, and offers a way to guide the simulation module to simulate in a proper product sequence.

Simulation Modeling Elements developed under the simulation engine, Simphony, are made to resemble the corresponding physical or logical real world elements. The derived instances of simulation modeling elements are used to assemble the simulation model, and the library that acts as the template is created based on these instances. Simulation modeling elements supplied in the system cover almost all demands for deriving the necessary instances, such as resources, construction methods, and conditional nodes, satisfying the requirements of the civil construction simulation.

The Simulation Product Model (ActiveX), an OLE object, works as the liaison and has the capability to link with all other modules in the CAD-ISE, which will be discussed later. It takes advantage of the ActiveX technology to communicate and control the outside object that develops separately, such as CAD-ISE and Simphony. Fig. 5-2 demonstrates the prototype system architecture.

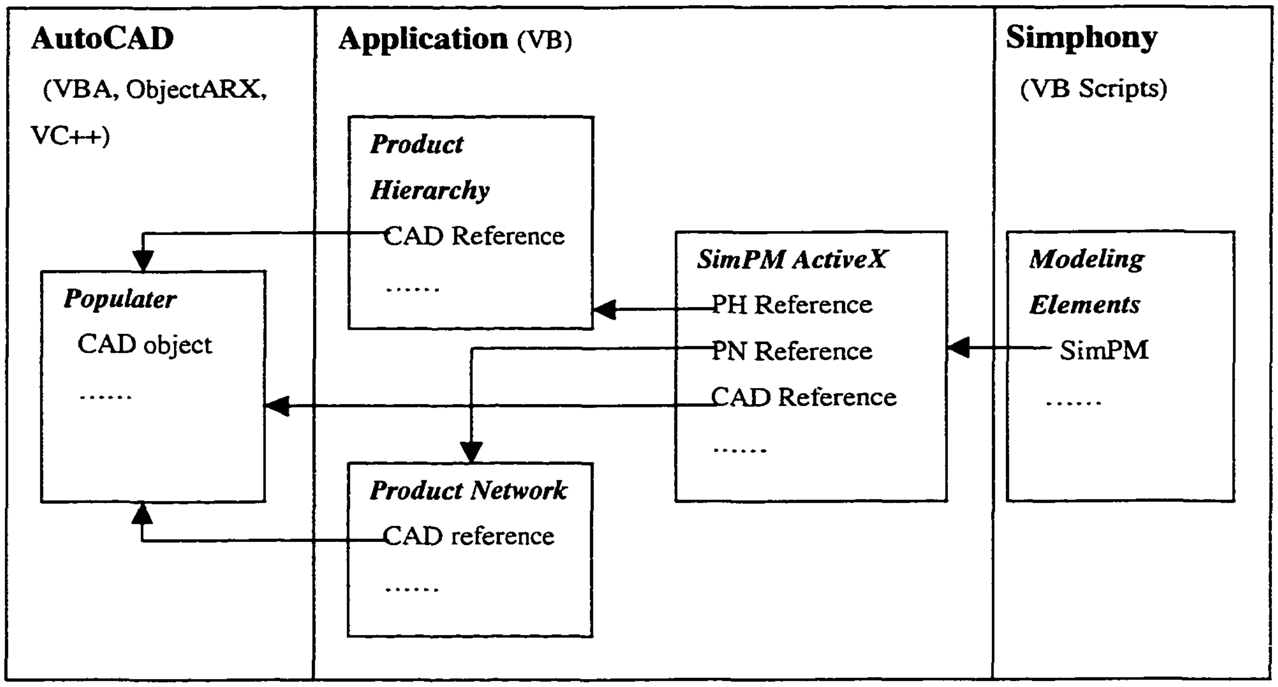


Fig. 5-2 CAD-ISE Developing Architecture

Link with Simphony. SimPM (ActiveX) is embedded in Simphony, and the SimPM instance can be created in the Simphony Designer. When the developer creates the entity modeling element in Simphony Designer, a new SimPM instance must be created simultaneously with the reference for the "PM" attribute of the entity set to this new instance. Thereafter, the SimPM object can be accessed through the "PM" attribute of the entity.

Link with product model and product network. Each project has only one product model, but it may have more than one product network, a factor that depends on how many construction phases are needed in the project. Each construction phase will be assigned one simulation model and one product network. Once the simulation model has been assigned to the construction phase, the references for the product model and product network associated with this construction phase will be passed to the simulation modeling element as attributes of the modeling element. The SimPM instance will queue these references later to link to the product model and product network.

Link with CAD. SimPM (ActiveX) also has a reference that is used to point to the CAD object. During the course of simulation, the model can send messages through this reference to communicate with the CAD program.

CAD-ISE provides the capability to conduct multiple construction phase simulation. In other words, multiple independent simulation models will be included in one simulation project. For example, pipeline construction may be divided into two construction phases: excavation, which can start from more than one place, and installation of pipe, which can only start from a certain point. These two construction operations may have different construction sequences. Two separate simulation models would properly suit this scenario better than one, however, certain common information must be shared between the two simulation models. Furthermore, there are some constraints shared among them, such as only after a certain length of the pipeline is ready (the first phase) can installation of the pipe start (the second phase). The SimPM object also plays a critical role in providing communications capability to solve the above problems.

The SimPM object acts as the message carrier that operates throughout the entire construction operation. Although the SimPM object is not the same for different construction phases, it has the same functionality and can pass messages effectively to provide the integration of the systems.

At the start point of the simulation, a SimPM object and the first entity are created, and an update message is sent to the source product model and source product network to update their information.

After the entity reaches the destination and finishes the last operation, an update message will be sent to the destination product model and the destination product network to update the information in these two objects. At the same time, another message will be sent to the following simulation model (if applicable) to fire the Check Product model event that determines whether there is enough product remaining to be worked on.

During the course of the simulation, the SimPM object continues sending messages to update the source product and destination product until the job has been completely finished or has reached other thresholds.

Take an earthmoving simulation with two construction phases as an example. Phase 1 is the moving of the dirt from source to destination and Phase 2 is the spreading and compacting of the dirt. Therefore, Phase 2 is constrained by Phase 1. Whether and when Phase 2 can start depends on Phase 1's progress. The following are typical messages that provide a means of integrating these two separate phases (simulation models) in the earthmoving simulation model (Fig. 5-3).

1. Message 1 is sent by the first activity through SimPM in Phase 1 after the SimPM is created along with the first entity, checks the PH to see if there is enough dirt available to be moved, then updates the information in the PH.
2. Message 2 is sent to the PN, which decides which PAC or PACs will be moved, and updates the status of each PAC in the PN.

3. Message m+1 is sent to update the information in the PH.
4. Message m+2 is sent to the destination PN, which determines where the PAC or PACs should go, and updates the status of each PAC in the PN.
5. Message m+3 is sent to the second phase (or model) to wake up the entity or entities that is/are waiting for the PAC or PACs (e.g. dirt).
6. Message m+4 is sent by the first activity through SimPM in the second phase to check the destination PH as to whether or not there is enough product (e.g. dirt) available, and updates the PH's information.
7. Message m+5 is sent to the destination PN to get the construction sequence information and update the status of each PAC in the PN.

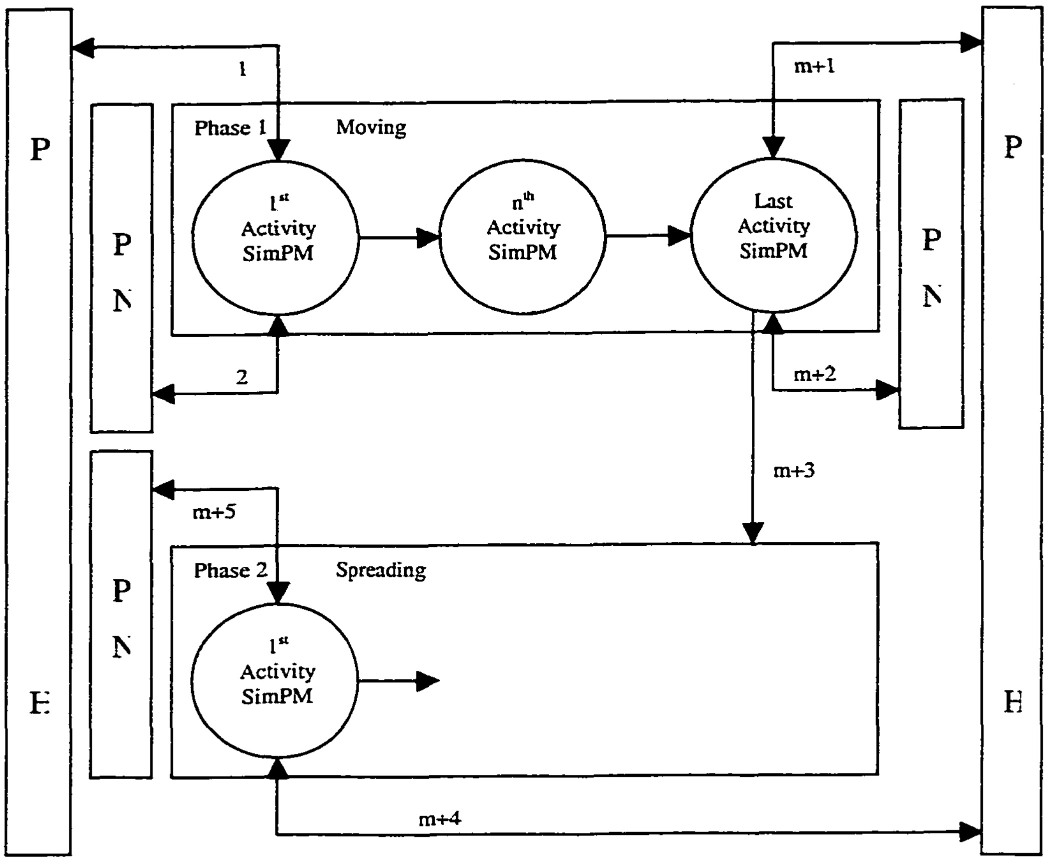


Fig. 5-3 Example of Message Flow

5.3 Development of the System Module

To satisfy the functionality required in the CAD-ISE, as outlined in section 4.2, the five modules are implemented. The relevant issues and implementation of each module are discussed in detail in the following sections.

5.3.1 Object-Oriented Approach

The concept of object-oriented simulation has great appeal in simulation applications because it is very intuitive to view the model of a system as composed of objects. This is especially true for simulation, where a system itself is viewed as a collection of interacting objects (Pritsker 1997). The CAD-ISE draws heavily on the object-oriented approach because, in addition to the natural one-to-one relationship between system objects and their computer-based representations, the CAD-ISE focuses on the product that is to be built and its components. It allows users to relate in a very natural way via discrete objects and offers solutions to construction industry simulation because of its ability to accurately represent the physical objects built during construction. The CAD-ISE also draws on the concepts of hierarchy and inheritance that are the major advantage of the object-oriented approach in the generation of a simulation model based on the product-oriented approach.

For anyone who wishes to simulate a construction project using a computer, the intricate relationship between the different components of construction presents a formidable challenge. It is difficult to develop a system that can behave realistically under all scenarios. One thing that a developer can do to alleviate this problem is to divide a complex system into its major parts. In other words, they can use an object-oriented approach where all components can be defined as objects that interact with each other and with the user. The developer can then piece together the components based on the construction sequence.

The following is an overview of some of the important object-oriented approach concepts used in developing CAD-ISE.

Classes and objects. An object consists of a collection of properties (data or information) and a set of methods that represents the object's behavior. Methods can access and modify an object's properties. A class is a generic definition of an object, and each object is an instance of its respective class.

Abstraction. Classes of objects are defined in terms of the features that are of importance in the context of the system being modeled. These definitions form abstract representations of the objects rather than their actual physical representations.

Encapsulation. The property of being a self-contained unit is called encapsulation, a feature that allows for data hiding. Data hiding is the highly valued characteristic that an object can be used without the user knowing or caring how it works internally, allowing the user to focus instead on how to use it.

Inheritance. The object-oriented approach supports the idea of reuse through inheritance; that is, a new type, which is an extension of an existing type, can be declared. This new subclass is said to derive from the existing class and is sometimes called a derived class. The derived class automatically inherits the entire instance properties and methods of the superior class and can add its own properties and methods that will extend to any of its derived classes.

Polymorphism. Polymorphism facilitates the definition of flexible entities that may refer to different classes, all of which offer an operation with the same external specification but with different implementations. The application of an operation to the entity will result in the appropriate implementation being selected; depending on the particular object associated with the entity at the time, the operation is executed.

5.3.2 Populater Module

The Populater Module is implemented under the CAD environment to capture and populate CAD information. The development of this module involves two application systems, AutoCAD and Eagle Point, and two development languages, ObjectARX and VBA.

AutoCAD is widely used by design engineers and it provides 3D support for all designed projects. AutoCAD Release 14 software delivers enhancements in areas with the most influence on user efficiency: improved performance, smart drawing tools, high-quality presentation features, and timesaving reference, raster, and Internet file-sharing capabilities. Other improvements make the software easier to learn, manage, and customize to user-specific needs. ActiveX Automation support allows AutoCAD software to integrate more closely with other Windows programs and allows for user-customization using programming tools such as Visual Basic.

ObjectARX™ is the AutoCAD object-based development environment used to create object-based CAD applications. ObjectARX was previously known as ARX for AutoCAD Release 13, or the AutoCAD® Runtime Extension. It contains many new features and aggressive performance improvements.

VBA is used to develop the interface that provides a means to create a model interactively. Visual Basic for Applications is an object-based programming environment designed to provide development capabilities from within an application. The integration of VBA into AutoCAD provides an easy-to-use visual tool for customizing the system. The VBA Editor included with Release 14 is the same environment used by Office 97 applications; Release 14 developers must only learn to use the one environment. VBA enables developers to use AutoCAD Release 14 ActiveX Automation features and operates rapidly as it is hosted by AutoCAD and therefore does not have the associated overhead of calling out to a separate process.

Eagle Point is being used to import the survey data, create 3D surface models, and aid in the creation of the final design surface. Two models are most important to the development of CAD-ISE: ***Surface Modeling*** is used to create a Triangulated Irregular Network (TIN) from a set of points and break lines in a drawing, or create contours or rectangular grids; ***Site Design*** is used to analyze and design every type of earthwork project, including landfills, parking lots, building pads, reservoirs and drainage ditches. Powerful surface-to-surface volume calculations make this module indispensable to this research.

There are two main issues for the data populater: the first issue is how to extract information from a CAD drawing; the second issue is to determine which processes will be used to populate the data. The following sections will discuss these two issues in detail.

5.3.2.1 Data Extraction

Extraction of CAD data is the key to achieve the integration of simulation with CAD. Since the CAD environment was designed for engineering and drafting purposes, it is not necessarily an ideal environment for simulation definition. However, the data for simulation definition exists in the system; the task becomes how to extract this data and use it.

The extractor module, developed in ObjectARX, is an add-on AutoCAD program (for the source code, refer to Appendix 3). The reason behind using ObjectARX to develop this module is that only ObjectARX can access all of the data stored in the CAD object. This module reads all necessary data from the CAD object data, such as a 3D-surface model object created by Eagle Point, and then writes this data to the extended partition of the same object where VBA can access it (Fig. 5-4).

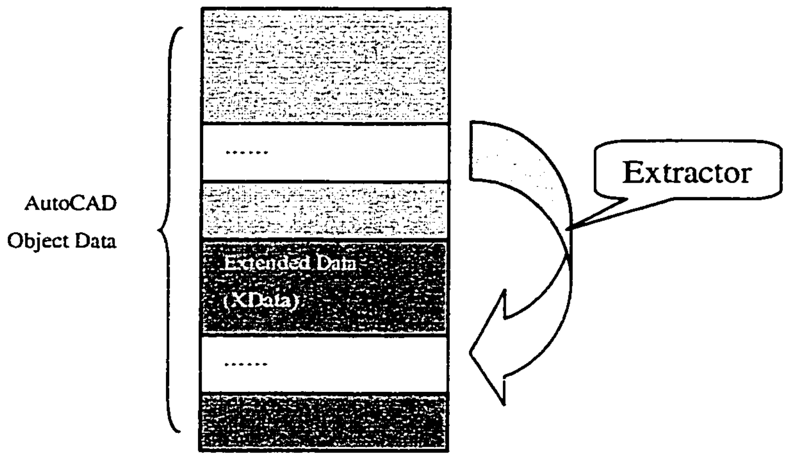


Fig. 5-4 Extractor Module

This module is simply a tool that makes data stored in CAD objects accessible to VBA. If the new VBA version is able to access the data in the CAD object directly, this module will not be needed any more.

5.3.2.2 Data Populater

The need to take into consideration detailed aspects of both design and the physical site introduces the complexity of populating this information for the simulation. Visual Basic for Applications in AutoCAD provides an ideal tool to develop a specific user interface that turns the platform into a tool for defining the PACs for simulation purposes.

Taking earthmoving as an example, the following describes how to achieve this functionality.

Generally, an earthmoving site consists of an excavation source, a single placement area. Fig. 5-5 illustrates a typical excavation site, made up of an original surface, sub-surfaces, working surfaces, and a final or design surface.

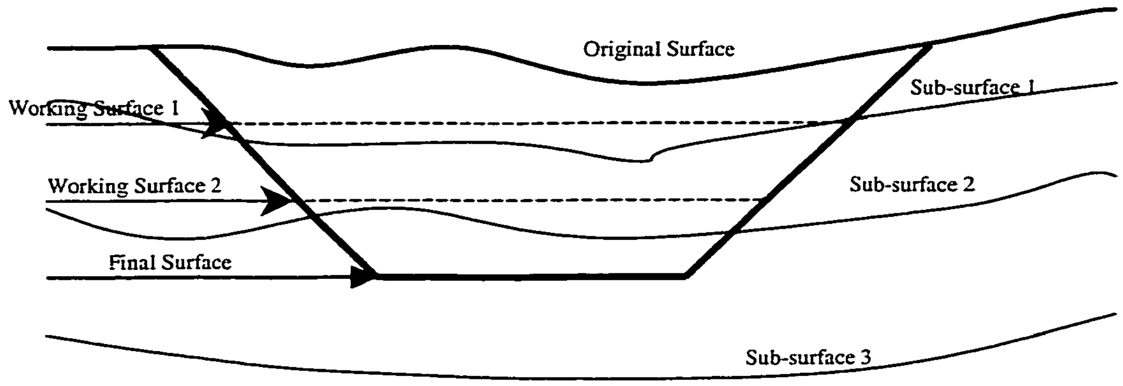


Fig. 5-5 Typical Excavation Site Cross Section

The earth enclosed within the final surface forms the product that the simulation model will simulate. Representing the product takes several steps:

1. Identify each surface. All the surfaces in a CAD drawing are created in terms of a 3D-surface object in Eagle Point. A 3D face class, a collection of TINs, is developed to interpret and represent these 3D-surface objects (refer to Appendix 4). One instance of this class will represent one surface by extracting geometric data and attributes from CAD objects. All natural surfaces, including the original surface and sub-surface(s), are located in the different layers while they are created. The final surface is also positioned in a separate layer. All working surfaces are created in separate layers by

specifying parameters in two ways: 1) three 3D stake points forming a surface, or 2) elevation. The layer attribute is used to identify each individual surface. The TINs in each layer compose a surface – an instance of the 3D face class.

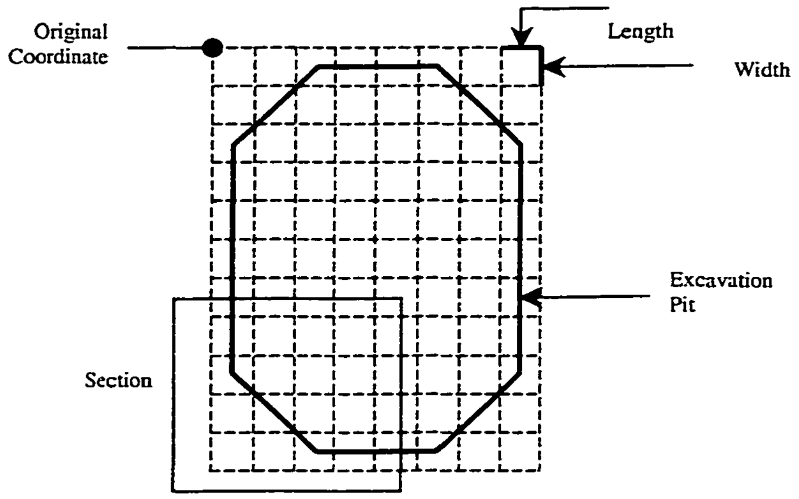


Fig. 5-6 Grids

2. Create a grid and calculate parameters. 3D-surface objects divide the product vertically. Then, a grid is used to divide the product horizontally by specifying the area and the grid width and length interactively (Fig. 5-6). A grid class (see Appendix 4) is developed to deal with the creating and storing of grids. Each grid's x- and y-coordinates will be calculated based on the area's original coordinates and the width and length of the grid. Two steps are taken to calculate the z-coordinate of each grid:

   1) Determine which TIN the grid is located in. First, project the 3D TIN (6ABC) and Grid Point O to the xy-plane to get 2D triangular 6A'B'C' and O' respectively (Fig. 5-7).

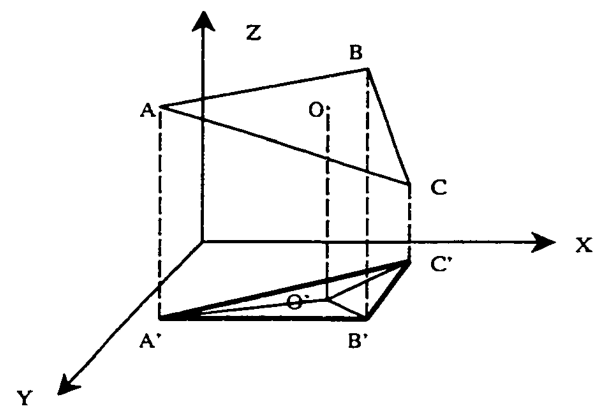


Fig. 5-7 Project 3D TIN to 2D-Triangular

IF

$$AREA_{6A'O'B'} + AREA_{6B'O'C'} + AREA_{6C'O'A'} = AREA_{6A'B'C'}$$

Then

Grid Point O in the TIN ABC

Equ. 5-1

2) Calculate the z coordinate. The equation of a plane passing through 3 points (x_1, y_1, z_1), (x_2, y_2, z_2), (x_3, y_3, z_3) is given by:

$$(x_3\ y_3\ z_3) \quad \begin{vmatrix} x - x_1 & y - y_1 & z - z_1 \\ x_2 - x_1 & y_2 - y_1 & z_2 - z_1 \\ x_3 - x_1 & y_3 - y_1 & z_3 - z_1 \end{vmatrix} = 0$$

Equ. 5-2

Or

$$\begin{vmatrix} y_2 - y_1 & z_2 - z_1 \\ y_3 - y_1 & z_3 - z_1 \end{vmatrix} (x - x_1) + \begin{vmatrix} z_2 - z_1 & x_2 - x_1 \\ z_3 - z_1 & x_3 - x_1 \end{vmatrix} (y - y_1) + \begin{vmatrix} x_2 - x_1 & y_2 - y_1 \\ x_3 - x_1 & y_3 - y_1 \end{vmatrix} (z - z_1) = 0$$

Equ. 5-3

Therefore

$$z = z_1 - \frac{\{[(y_2 - y_1)(z_3 - z_1) - (y_3 - y_1)(z_2 - z_1)](x - x_1) + [(z_2 - z_1)(x_3 - x_1) - (z_3 - z_1)(x_2 - x_1)](y - y_1)\}}{[(x_2 - x_1)(y_3 - y_1) - (x_3 - x_1)(y_2 - y_1)]}$$

Equ. 5-4

3. Create a CAD object and build a Product Hierarchy (PH). The 3D-solid box object is created to represent the smallest unit of product and corresponds to one PAC in the Product Hierarchy (Fig. 5-8). The PH is built by indicating the zone in the CAD object as a section in the hierarchy. The PACs will then be created to point to each 3D-box CAD object. The system will automatically generate the PH based on the information carried by each box (Fig. 5-9).

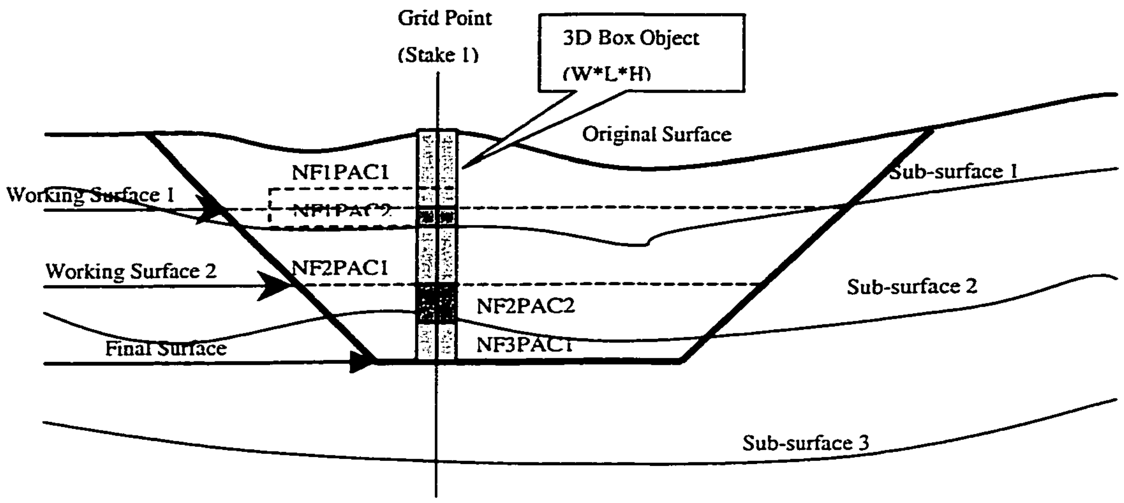


Fig. 5-8 PAC's CAD Representation

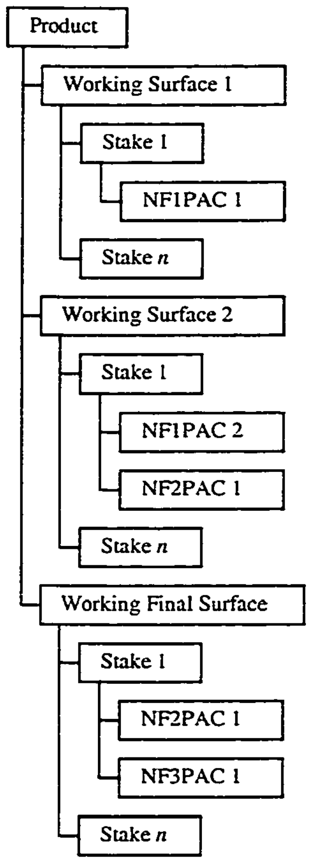


Fig. 5-8 PH

Although the placement site will not necessarily need to be defined in any detail, it could be divided and described in the same manner as the excavation site.

In order to represent a linear product, such as pipeline and road construction, a linear PAC module is also provided to automatically create a PH by drawing the path and giving the length and cross-section area (if applicable) of each PAC.

All of the functionality is implemented as an add-on program and is added to the AutoCAD menu. The user can use this functionality by accessing the menu items in the same way as regular AutoCAD menu items.

5.3.3 Product Hierarchy Module and Product Network Module

The Product Hierarchy module and Product Network module provide most of the functionality for the prototype system. The user interacts with these two modules using the graphical user interface developed for each module.

The purpose of the Product Hierarchy module is to create an object-oriented representation of the project product and establish relationships between all product components. The first step in this module is to generate PACs from the refined 3D CAD drawing and create the Product Hierarchy (PH). The 3D CAD data extracted from the AutoCAD application provides information such as component geometry and location for instancing the objects. The next step in this module is to integrate the PH with the simulation model. The module finishes its final setup by initializing the simulation model, which extracts the essential information, including the construction phase (corresponding to the simulation model), construction activity, and associated resources that are defined in each simulation model to initiate the corresponding objects included in the PH. The final step in this module is to update the property values of all objects where the value of the property is dependent on the simulation result and to collect the simulation results, such as activity start time, duration, finish time, and resources used, which will be used to generate the scheduling and estimating during the course of simulation.

The Product Network (PN) module offers the modeler a visual method of defining the Product Operation Sequence (POS). Operation sequencing allows the modeler to specify the implementation strategy for the project being planned. This information is important in determining the implementation sequence of the underlying processes.

Based on the construction logic, section, technology, strategy, and resource availability, various phases can be sequenced in different ways.

To implement these two modules, the following classes are developed (also see Appendix 4).

5.3.3.1 CADISE_Simphases Class

The CADISE_Simphases class is a collection of CADISE_Simphase objects dealing with construction phases. Besides collecting all construction phases associated with the product, CADISE_SimPH, which represents the product, is also included in this class. To instance this class creates product simulation object which has all the information required to conduct a simulation.

5.3.3.2 CADISE_Simphase Class

The CADISE_Simphase class provides the definitions for every individual construction phase by instancing this class. This class contains the subclasses CADISE_Activity, BridgeX (ActiveX), and CADISE_SimPN, which will be discussed later.

5.3.3.3 CADISE_SimPH Class

The CADISE_SimPH class is used to represent the Product Hierarchy structure. The instance of the NetTree class provides the visual definition of the product and the NetNodes class provides the internal definition of the product that could be a single or a group of PACs.

5.3.3.4 NetTree Class

NetTree class is a graphic representation used to represent the Product Hierarchy and Product Network visually. It provides basic functionality to accomplish data entry interactively, including the creation of the PH and definition of the construction operation sequence. The polymorphism of object-oriented methodology plays critical role in this class, as it determines the behavior of the class at run time. The property of the NetTree class defines the object style and events, customizing the behavior of the object.

5.3.3.5 NetNodes Class

NetNodes class is an internal representation of the product in different forms, depending upon which level it locates. For example, if it is in the lowest level of the product tree, it represents only one PAC; otherwise, it represents a group of PACs. The properties and behaviors also apply to an individual PAC or a group of PACs.

5.3.3.6 acadPAC Class

The acadPAC class is used to derive the CAD object that points to the 3D-box object in the CAD drawing. This object creates a linkage between the CAD program and the simulation model. The properties of this object provide information to define the PACs, such as volume, type, and quantity.

5.3.3.7 CADISE_Statuses and CADISE_Status Class

CADISE_Statuses class is a collection of CADISE_Status objects that represent the status of each phase, including CanStart, IsStarting, and HasDone, which describe each NetNode's status. The Activities object, which is derived from the

CADISE_Activity class and provides a definition of resources, is used to conduct certain construction methods (activities) and stores data regarding completion activity used to generate schedule and estimating.

5.3.3.8 CADISE_Resource Class

CADISE_Resource class handles resources required to accomplish specific tasks. It contains crews, materials, and others. The crews can be divided further to include labour, equipment, and rental.

5.3.3.9 CADISE_SimPN Class

The CADISE_SimPN class defines the product construction sequence. The major component is NetTree, mentioned before, which provides a graphical interface to indicate the successor(s) and predecessor(s) of an object.

5.3.3.10 BridgeX Class

The BridgeX class is compiled to be an ActiveX, which builds a bridge between the main model and SimPM, another ActiveX class declared in Simphony by using VB script.

Fig. 5-10 illustrates the major classes and relationships between them in SimCAD.

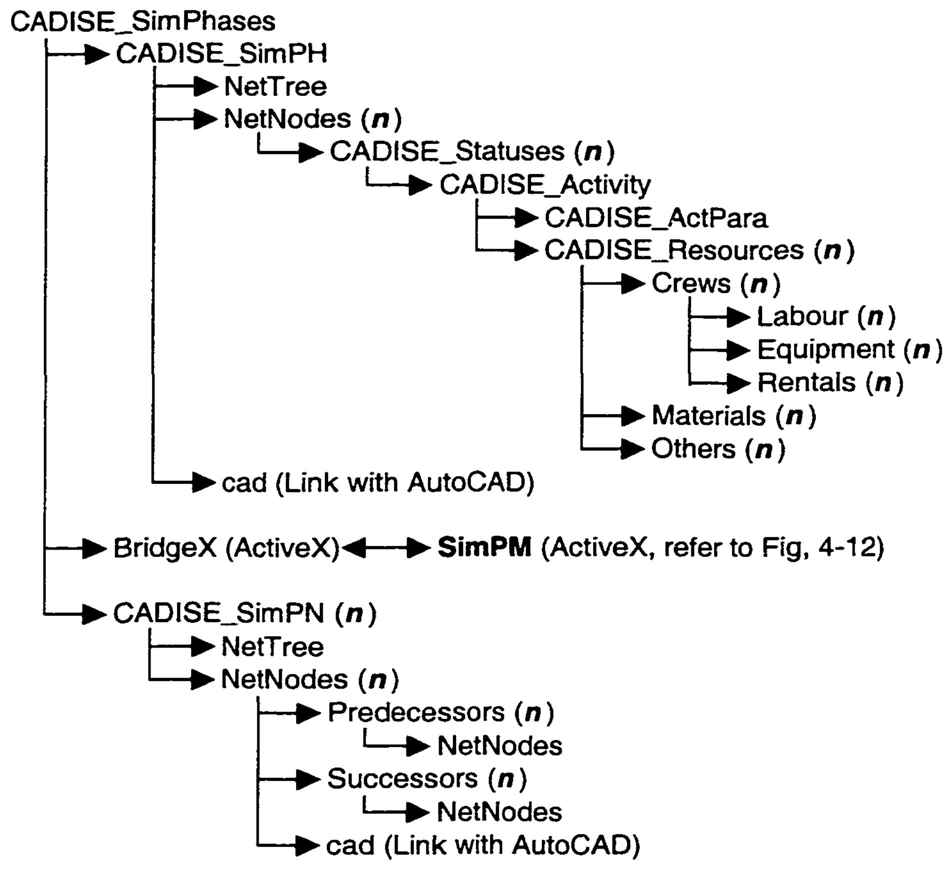


Fig. 5-9 SimCAD Class Map (see also Appendix 4)

5.3.4 Simulation Model Module (SimPM) - ActiveX

A form of "intelligence" should be built into the modeling environment to facilitate efficient model building. Design-specific information should be incorporated directly into the model through interfaces with CAD, supported by rule-based expert systems and object-oriented modeling.

The first step in integrating the CAD model with Simphony (simulation model) is to create the object that will serve as the liaison. The object must not only work closely with the simulation model, but also access the Product Model and Product Network created outside Simphony. In order to meet this requirement, the Simulation Product Model (SimPM) is developed. SimPM is developed independently and will be added into Simphony as the CAD extension using ActiveX technology. The integration between SimPM and Simphony is implemented through the Simphony Designer that acts as the developer interface.

A SimPM object defined as an instance of the SimPM class is declared in Simphony using VB script. Defining a new variable as a type of object in the entity modeling element and referring to the SimPM object effectively integrates the simulation model and product model. Communication between the Product model and simulation model is also achieved through this essential integration. Fig. 5-11 illustrates the SimPM class structure and its relationship with other modules.

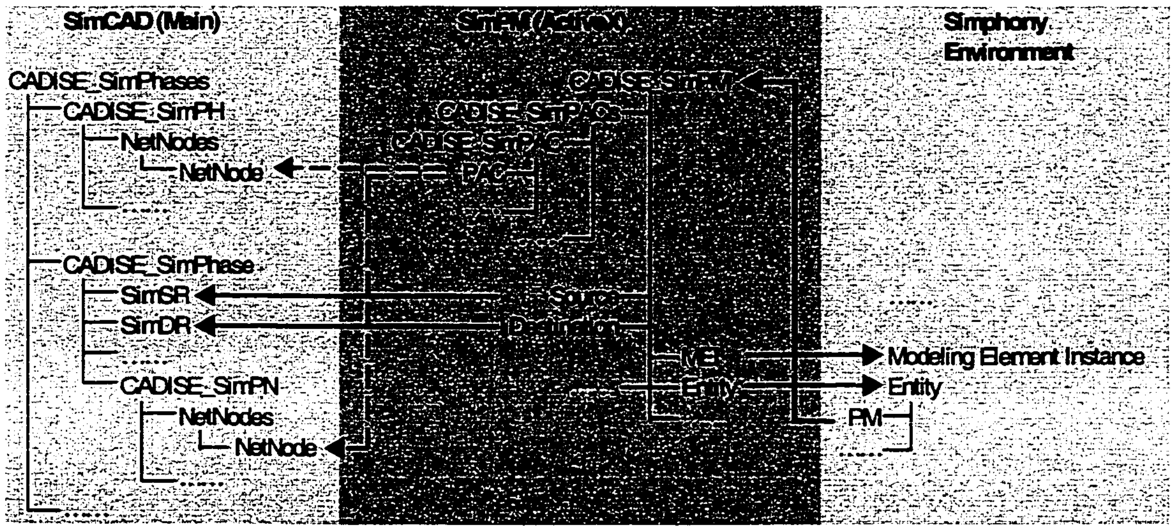


Fig. 5-10 SimPM Class Structure and Linkage

Each entity in the simulation model will carry a SimPM instance to track the entity, while simultaneously sending information to the PH and PN. The entity defined in the simulation model could also have a cycle. For instance, a truck as an entity in an earthmoving simulation moves dirt from a source to a placement and returns to the source again after dumping the dirt. The SimPM instance, acting like a piece of the product itself, travels from source to placement. However, once it proceeds through the entire simulation model, it will be destroyed. A new SimPM instance will be created after the entity (truck) returns to the source and begins moving another amount of dirt.

5.3.5 Simulation Engine

CAD-ISE utilizes an internal simulation engine, i.e., Simphony, developed by the Construction Engineering and Management group at the University of Alberta, which provides an object-oriented environment and supports all features of simulation.

5.3.5.1 Simphony

Simphony, a simulation engine used by POSE, is an event-scheduling simulation queuing model that arose as an object oriented implementation for special purpose simulation. It provides run-time support and repeatedly activates execution. The event-driven algorithm simply scans the module for an event that could possibly be scheduled for execution. The event selection is based on resource availability, logical constraints, other module constraints, and random phenomena. The selected event is routed to the appropriate process where future events can be scheduled (AbouRizk, Hajjar 1997).

In addition to event scheduling, it also provides numerous other forms of support, such as a low-level modeling element designer that allows the advanced user or developer to develop special modeling elements by using VB-scripting, a high-level simulation model editor that allows users who do not have a simulation background to build the simulation model based on the built-in modeling elements and pseudo-random number generation.

5.3.5.2 Modeling Elements

An interactive graphical environment that builds the simulation model in a natural and logical manner was deemed a major factor in the success of any simulation tool. In Simphony, this goal was achieved using object-oriented concepts and event-driven programming in the implementation process. A simulation model is developed by instantiating (derivation of an instance of a pre-defined object) the modeling elements provided in CAD-ISE. Internally, these modeling elements have been implemented as "objects," and have two representations: "Graphic" and "Simulation". The graphic representation of these objects assists in the operation of the graphical user interface while the simulation representation assists in the development of simulation models. The modeling elements were developed in an object-oriented programming environment, Simphony, with VB script. With this strategy, properties of the modeling elements were specified by the end-user to produce the desired behavior. Fig. 5-12 illustrates this concept graphically. This strategy was found to be suitable for the prototype where the construction modeling aspects were more important than the program's efficiency.

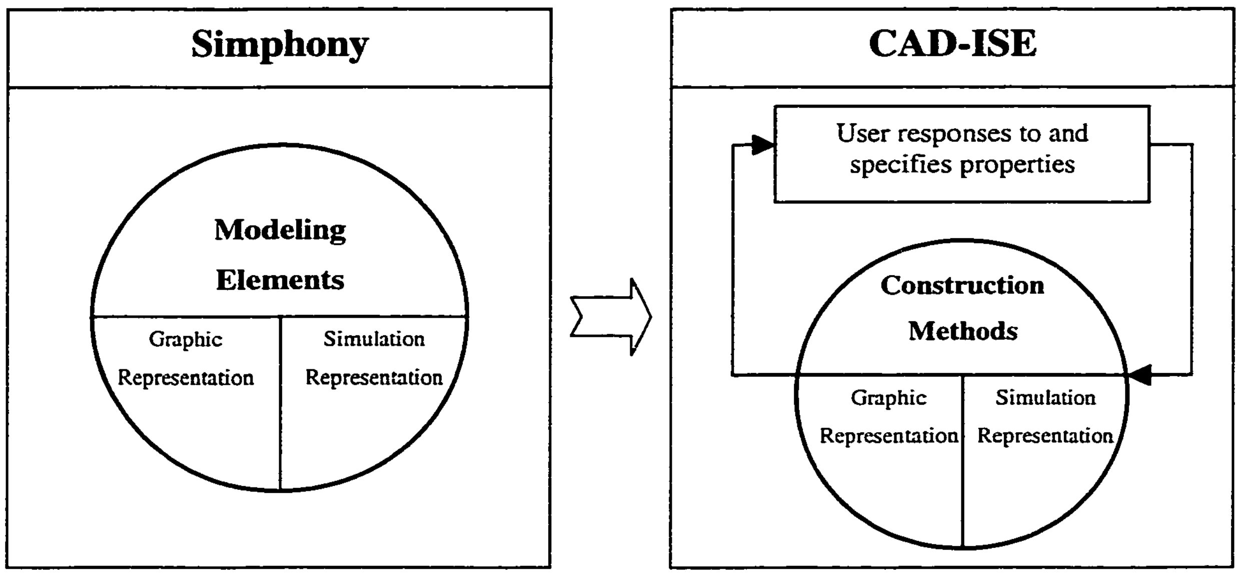


Fig. 5-11 Modeling Elements vs. Construction Methods

Table 5-1 provides a list of the major CAD-ISE modeling elements with a brief description.

Table 5-1 CAD-ISE Modeling Elements

Element	Description	Parameters
Branch	Implements probabilistic branching	Image, Name, Probability
Destroy	Destroys incoming entities	
Entity	Creates simulation entities carried from product-based entities (PE)	Number of Entities, Time of First create, Time between create, Image, Name, Descriptive Type, Process Priority, Product Mode, Capacity, Tolerance
Activity	Defines a construction method or a set of construction methods	Image, Name, Activity ID, Governing Activity, PM Created from, Site Number, Update PH, Statistic Collection, Activate Sub. Activities, Activity Quantity, Activity Productivity, Activity Duration
EntityIn	Routes entities arriving at an input connection point of the parent element	
EntityOut	Routes entities to an output connection point of the parent element	
Filter	Implements conditional branching	Entity Descriptive Condition, Site, Working Space, Section, PAC Type
Buffer	Creates working space buffer	Number of the Buffer
Dummy	Creates dummy entity that is sent back to fire the waiting entities	
Clone	Duplicates simulation entity to form a cycle	
Crew, Labour, Equipment, Rental, Material, Other	Registers resources for modeling elements	Image, Resource Name, Total Number of Resources
Resource	Requests resource for activity or entity	Image, Resource Name, Number of resources to request, Resource Priority, Allow Holding, Allow Interrupt

Without extensive programming knowledge, the user can fully customize and create new libraries of modeling elements. These product-oriented modeling elements used for modeling the construction project resemble the way the project is dealt with in the real world.

Action logic is used throughout a model to control the events in the process. CAD-ISE allows a modeler to define scripts similar to a simulation language using statements and expressions. The more the user understands simulation and VB language, the more he/she can customize the modeling elements.

The product-oriented modeling elements are built on top of the simphony modeling elements that inherit all properties and behaviors from simphony modeling elements. In addition, the new properties and behaviors dealing with the product model are implemented into them to achieve the integration with the product model. The following explains some modeling elements that play a critical role in product-oriented simulation.

5.3.5.2.1 Entity

CAD_Entity, an entity modeling element, is used to create a simulation entity (see Chapter 3) with the ability to carry a product-based entity (e.g. SimPM). Besides the common functionality that supports normal activity-based simulation, the entity created by CAD_Entity enhances its ability to communicate with the product model through the following additional features.

A specific object variable is created to reference SimPM, an ActiveX represented product model, when the entity is initiated. The *Descriptive Type* parameter lets the user specify the type of entity that is going to be created using a descriptive word, such as "small" or "large" to indicate a different type of entity, which will be used by the filter modeling element to determine which branch will be taken. The *Process Priority* parameter formulates the processing order among the waiting entities. The method used to measure product, such as volume, piece, area or length, is determined by the *Product Mode* parameter. *Capacity* and *Tolerance* provide information to guide product model creation later.

Although SimPM is initialized while creating the entity, it is only referenced to the specific PAC after it processes through the activity modeling element and fires the create event discussed in Chapter 4.

5.3.5.2.2 Activity

Activities typically represent processing events. The detailed definition of product-based activity is given in Chapter 3. The implementation of the CAD_Activity modeling element is discussed in the following section.

In order to accomplish the functionality described in Chapter 3, several properties are introduced and the behavior of each instance of the activity modeling element is implemented through these properties. The *PM Created from* property guides the entity that is passing through the activity where the PM was created. It has three options, *None* – no PM is created; *Source* – PM is created from source PH; or *Destination* – PM is created from destination PH. The *Site Number* property provides a way to indicate which site the product model will come from or go to, dealing with multi-source site and multi-destination site situations. The *Update PH* property controls where the update message will be sent: if the value is source, the source PH will be updated; if both, the source PH and destination PH will both be updated; if none, none of them are updated. The

Governing Activity property determines whether this activity will affect the succeeding construction phase operating on it. For example, dumping is usually a governing activity, which means that only after the current PM finishes its dumping can the following construction phase (spreading) begin operating on it. However, the returning activity is a non-governing activity, and it does not affect the spreading phase. The *Activate Sub. Activities* property indicates whether there is a sub-model combined pair of activities to simulate this activity in more detail. This feature makes building a construction method representation hierarchy possible.

5.3.5.2.3 Filter

The CAD_Filter modeling element provides a way to filter the different types of entities based on their built-in properties. In other words, an instance of the CAD_Filter modeling element can be used to pick out those entities that meet the condition set by the filter. *Entity Descriptive Condition* uses the entity's descriptive type, such as "small" or "large," to set the criteria; *Site, Working Space, Section,* and *PAC Type* use the information carried by the PM to decide which path it will take. This modeling element provides a means of accomplishing a multi-site situation, having different PAC types moving to different site situations, different entity types moving through different activity situations, and so on.

5.3.5.2.4 Crew, Labour, Equipment, Rental, Material, and Other

All these modeling elements are used to define the resources required to perform a construction activity. They register the resources, which can then be requested by the instance of the CAD_Resource modeling element, which will be discussed later. The associated classes organize the resource structure and link to the resource database, which provides the information needed to perform the estimating (see Appendix 4).

5.3.5.2.5 Resource

The CAD_Resource modeling element requests construction resources, such as crew, labour, equipment, and material. The instance of the CAD_Resource modeling element can be used to request different types of resources, and the resources requested may be of crew, equipment, or material types. The instance of the CAD_Resource modeling element only can be located in the entity and the activity modeling element, and it can access the resources registered in the same or higher-level elements. Properties, including *Resource Priority, Allow Holding,* and *Allow Interrupt,* govern the way that the resources are allocated.

The following rules are applied to allocate the resources.

1. If all requested resources are available, capture the resources and start the activity.
2. If more than one activity requests the same resource(s) at the same time, the activity with the higher priority gets the resource(s).
3. If the activity requests multiple resources and they are not available at the same time, the following rules are followed:
    1) If the *Allow Holding* setting is true, allocate and hold the resources that are available currently;
    2) Otherwise, the resources can only be allocated when all requested resources become available.

The CAD_Resource modeling element is only responsible for setting the rules for requesting, allocating, and releasing; procedures are implemented in the Activity modeling element.

5.4 Conclusion

The prototype system developed fulfills the major functions of a product-oriented simulation methodology. First, it demonstrates one of the many roles that product-oriented simulation methodology can play in an integrated simulation system while demonstrating the successful integration of CAD with simulation through a product-oriented methodology. Second, the prototype system illustrates the way in which an external, stand-alone program can tie into an overall integrated system through the ActiveX mechanism. Third, the prototype system addresses many issues encountered in implementing product-oriented simulation methodology for these solutions.

CHAPTER 6 EARTHMOVING SIMULATION USING CAD-ISE

6.1 Introduction

Equipment intensive operations are associated with the heavy and highway sectors of construction. Such construction is repetitive in nature, and its planning is based around the assignment of major equipment for the construction process. The main challenge in modeling such operations is to provide a modeling environment where the level of abstraction required is minimal and the resemblance to the actual operation is high. In approaching this problem, the model structure can be thought of in terms of equipment that must be linked to processes that characterize a given type of heavy construction (e.g. earthmoving). Construction processes must be studied and their properties identified as they relate to product-oriented modeling. Generic structures and simulation constructs must be developed to facilitate the implementation of construction processes into a product-oriented simulation environment.

A CAD-based product-oriented special purpose simulation template called "CAD Earthmoving," developed to demonstrate the product-oriented simulation concept, is presented in this chapter. Users are able to model preparation, loading, hauling, dumping, and spreading operations in a simple and straightforward manner using this template.

6.2 Earthmoving

Earthmoving is a specialized construction field where large amounts of earth are moved from one location (referred to as the source), to another location (referred to as the placement). Earthmoving projects consist of many interacting processes, such as preparing, loading, hauling, dumping, and spreading.

The allocation and optimization of the resources involved in earthmoving is quite time consuming, and estimating the cost and duration of an earthmoving project involves a large number of assumptions about the effect of environment and resources on the overall system production. Simulation offers an effective way to solve these problems. CAD-ISE can be used to develop a project model that can access the advanced capabilities of simulation with minimal additional effort and knowledge about the specifics of simulation.

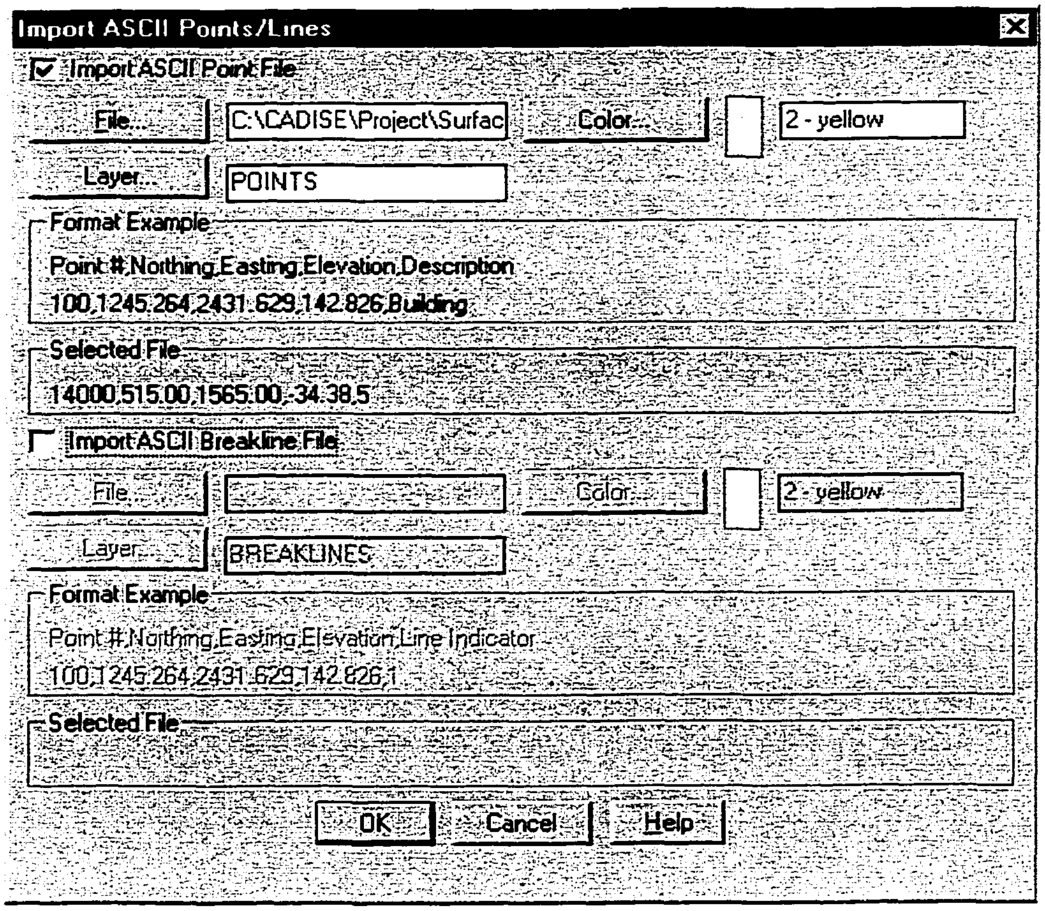
6.3 Earthmoving Product Model – PAC

6.3.1 Excavation Design

In order to understand the development of product-oriented simulation modeling for earthmoving under CAD-ISE, it is essential to have an understanding of the excavation design and the process in which it is created. The design of the site is created in AutoCAD with the aid of an add-on program, Eagle Point, which provides modules for road designs, surface modeling, site designs, data importing and exporting, survey layouts, and water and sewer designs.

The design of the excavation will result in a CAD file that includes definitions of each surface, including original surface, sub-surface(s) (if applicable), and final surface. Generally, three steps are taken to design an excavation site: importing data, creating natural surfaces, and designing final excavation surface.

- **Importing Data**. To develop surfaces in AutoCAD that represent the original surface and the different soil strata sub-surface(s), the surface elevation data must first be imported. The Eagle Point "Data Transfer" function is used to import raw elevation data from borehole logs and other sources (Fig. 6-1).


Import ASCII Points/Lines

[✓] Import ASCII Point File

File...: C:\CADISE\Project\Surfac
Color...: 2 - yellow
Layer...: POINTS

Format Example
Point #,Northing,Easting,Elevation,Description
100,1245.264,2431.629,142.826,Building

Selected File
14000,515.00,1565.00,-34.38,5

[ ] Import ASCII Breakline File

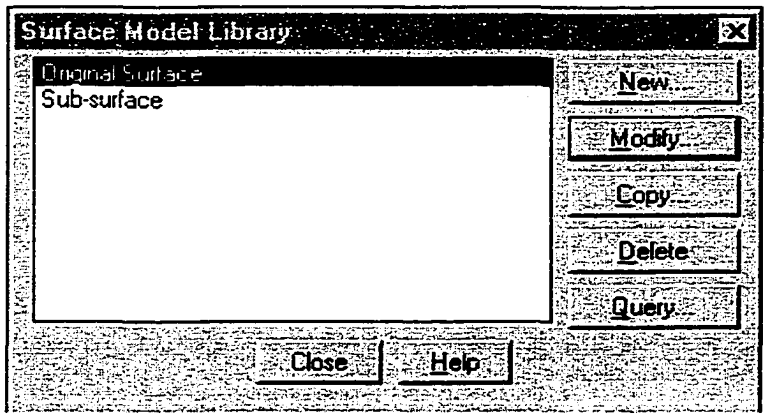
File...:
Color...: 2 - yellow
Layer...: BREAKLINES

Format Example
Point #,Northing,Easting,Elevation,Line Indicator
100,1245.264,2431.629,142.826,1

Selected File

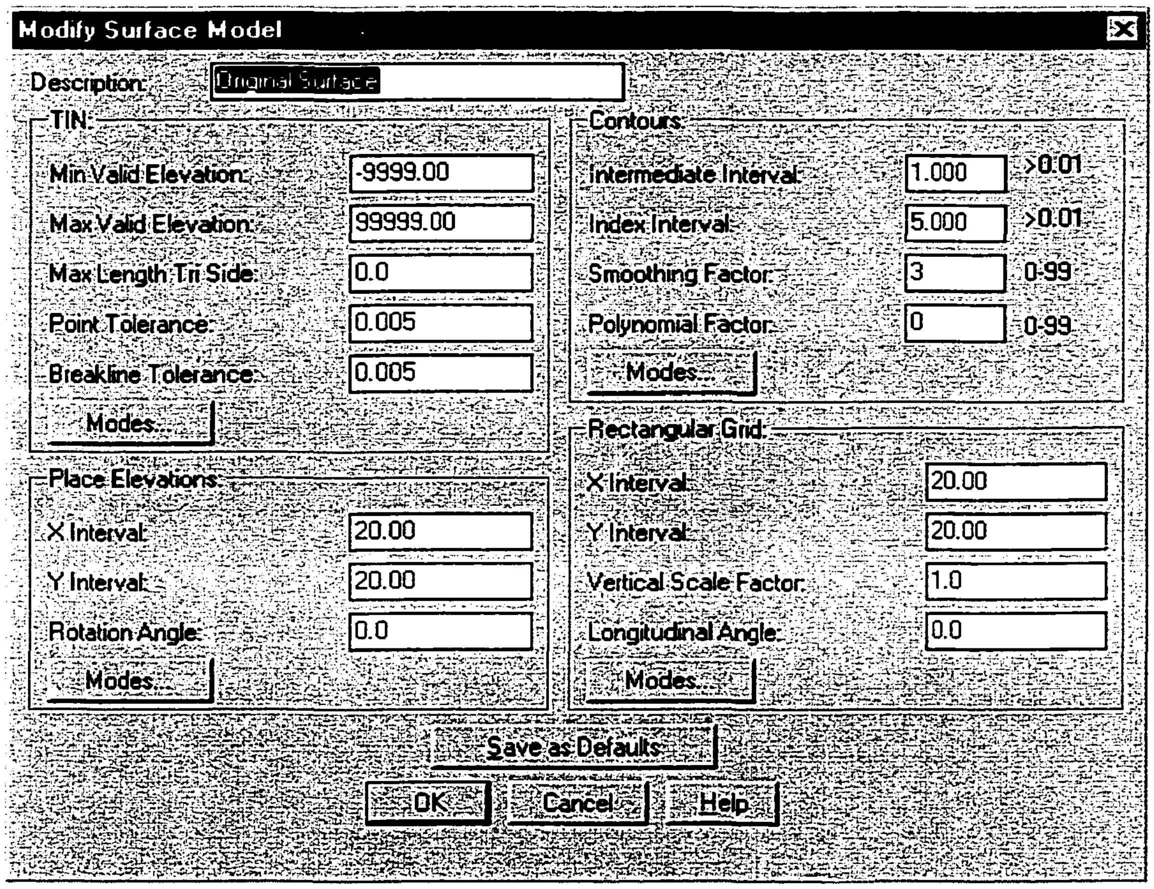
Fig. 6-1 Importing Data

- **Creating Natural Surface(s)**. Surfaces are created using the "Surface Modeling" function within Eagle Point (Fig. 6-2, Fig. 6-3). Surface modeling creates a surface by making a TIN (Triangulated Irregular Network) between points (Fig. 6-4). Each triangle acts as a plane between the points, creating a continuous surface over the area enclosed by the points. Each TIN that represents one soil strata layer or the original surface is stored in a "Surface Model Library".



Original Surface
Sub-surface

Fig. 6-2 Create Surface Model Library



Description: Original Surface

TIN:

Min Valid Elevation:	-9999.00
Max Valid Elevation:	99999.00
Max Length Tri Side:	0.0
Point Tolerance:	0.005
Breakline Tolerance:	0.005

Contours:

Intermediate Interval:	1.000	>0.01
Index Interval:	5.000	>0.01
Smoothing Factor:	3	0-99
Polynomial Factor:	0	0-99

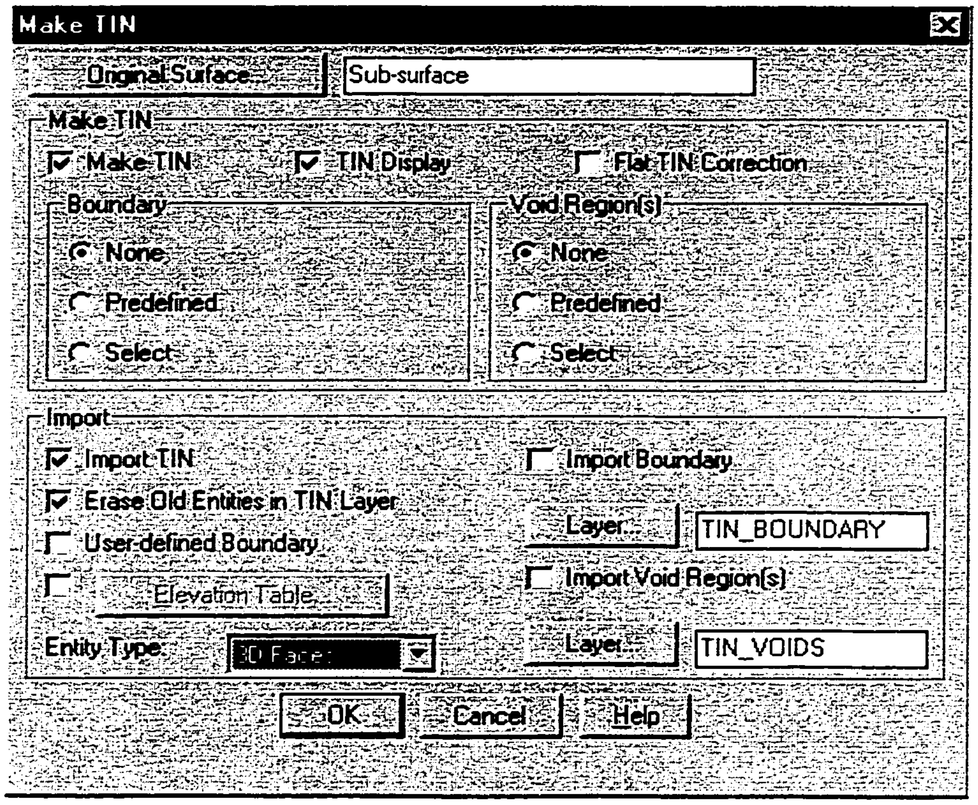
Place Elevations:

X Interval:	20.00
Y Interval:	20.00
Rotation Angle:	0.0

Rectangular Grid:

X Interval:	20.00
Y Interval:	20.00
Vertical Scale Factor:	1.0
Longitudinal Angle:	0.0

Fig. 6-3 Define Surface Model


Make TIN

Original Surface... Sub-surface

Make TIN
- [✓] Make TIN
- [✓] TIN Display
- [ ] Flat TIN Correction

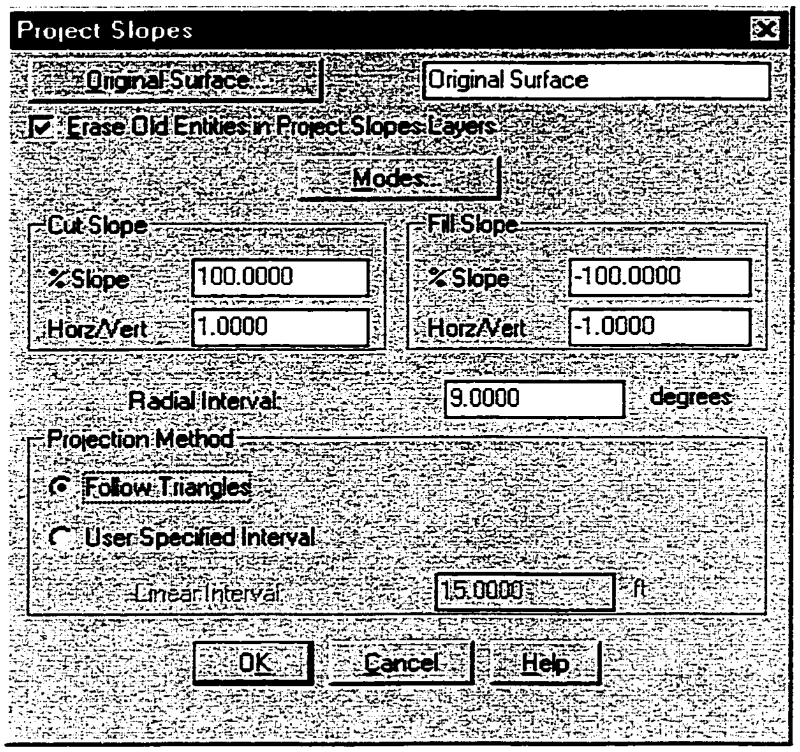
Boundary: [● None] ○ Predefined ○ Select

Void Region(s): [● None] ○ Predefined ○ Select

Import
- [✓] Import TIN
- [✓] Erase Old Entities in TIN Layer
- [ ] User-defined Boundary
- [ ] Elevation Table...
- Entity Type: 3D Face
- [ ] Import Boundary
- Layer... TIN_BOUNDARY
- [ ] Import Void Region(s)
- Layer... TIN_VOIDS

Fig. 6-4 Make TIN (Triangulated Irregular Network)

- **Designing Final Excavation Surface.** Designing the final excavation surface is typically done using a bottom-up approach. The final design of the excavation base and the slopes are all known. By drawing the desired excavation base outline, Eagle Point's "Site Design" module can be used to create slope lines from the base feature line to the original surface. The slope lines extend out from the feature line to the catch line, which results from the intersection of the slope lines and the surface (Fig. 6-5).


Project Slopes

Original Surface...: Original Surface

[✓] Erase Old Entities in Project Slopes Layers

Modes...

Cut Slope
%Slope: 100.0000
Horz/Vert: 1.0000

Fill Slope
%Slope: -100.0000
Horz/Vert: -1.0000

Radial Interval: 9.0000 degrees

Projection Method
[● Follow Triangles]
○ User Specified Interval
Linear Interval: 15.0000 ft

OK | Cancel | Help

Fig. 6-5 Design Final Surface

6.3.2 Product Model – PAC

Each surface, including original surface, sub-surface(s) (if applicable), and final surface, is represented by the surface model defined in Eagle Point. To create a Product Model – PAC based on these surface models, six steps are taken: interpreting the surface model, populating the data, creating natural face object(s), creating a final face object, creating working face object(s), and generating PAC objects (refer to Chapter 4 for concept detail).

- **Interpreting Surface Model.** The surface models created in Eagle Point must be transferred to common 3D face models, which can in turn be manipulated by VBA within AutoCAD. It is simply done by importing the surface model and creating 3D faces in a separate layer, where each layer represents one surface. The "Make TIN" form is also used for this purpose (Fig. 6-4).

- **Populating Data.** The *populator* add-on tool is used to extract information from the model by selecting and populating all 3D entities that make up the model.
- **Creating Natural Face Object(s).** Each natural face object is created by selecting all 3D faces located in one natural surface layer. A menu item called *Create Natural Faces* is used for this purpose.
- **Creating a Final Face Object.** All 3D Faces in the final surface layer form a Final Face object. The *Create Final Face* menu item offers this functionality.
- **Creating Working Face Object(s).** A working face object can be represented in two ways: as three points or as one elevation (Fig. 6-6). This tool is also found in the menu.

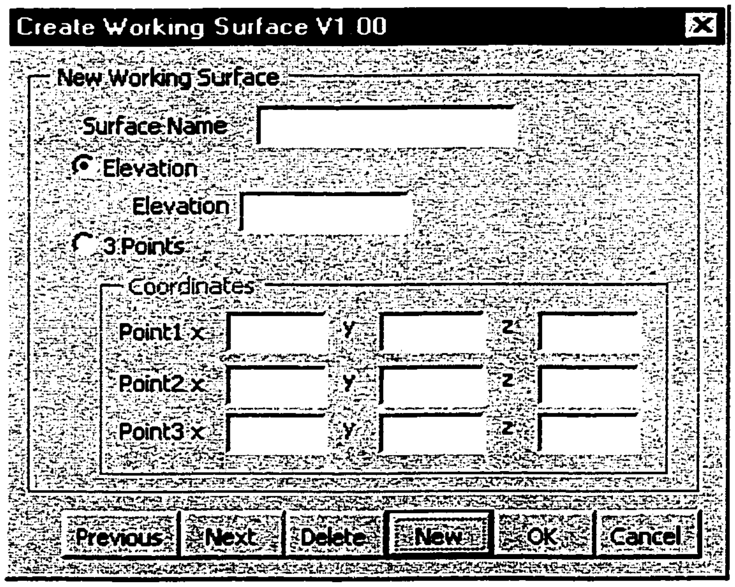


Fig. 6-6 Create Working Face(s)

- **Generating PAC Objects.** PAC objects are generated automatically through the *Create Grids* functionality located in the "Simulation Tools" menu (Fig. 6-7). Fig. 6-8 illustrates the entire project broken into PACs.

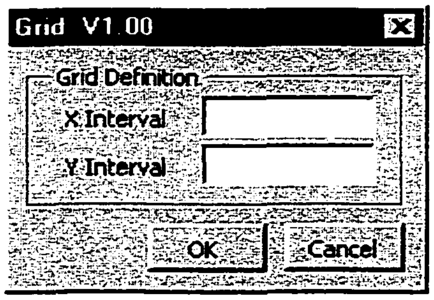


Fig. 6-7 Create Grids

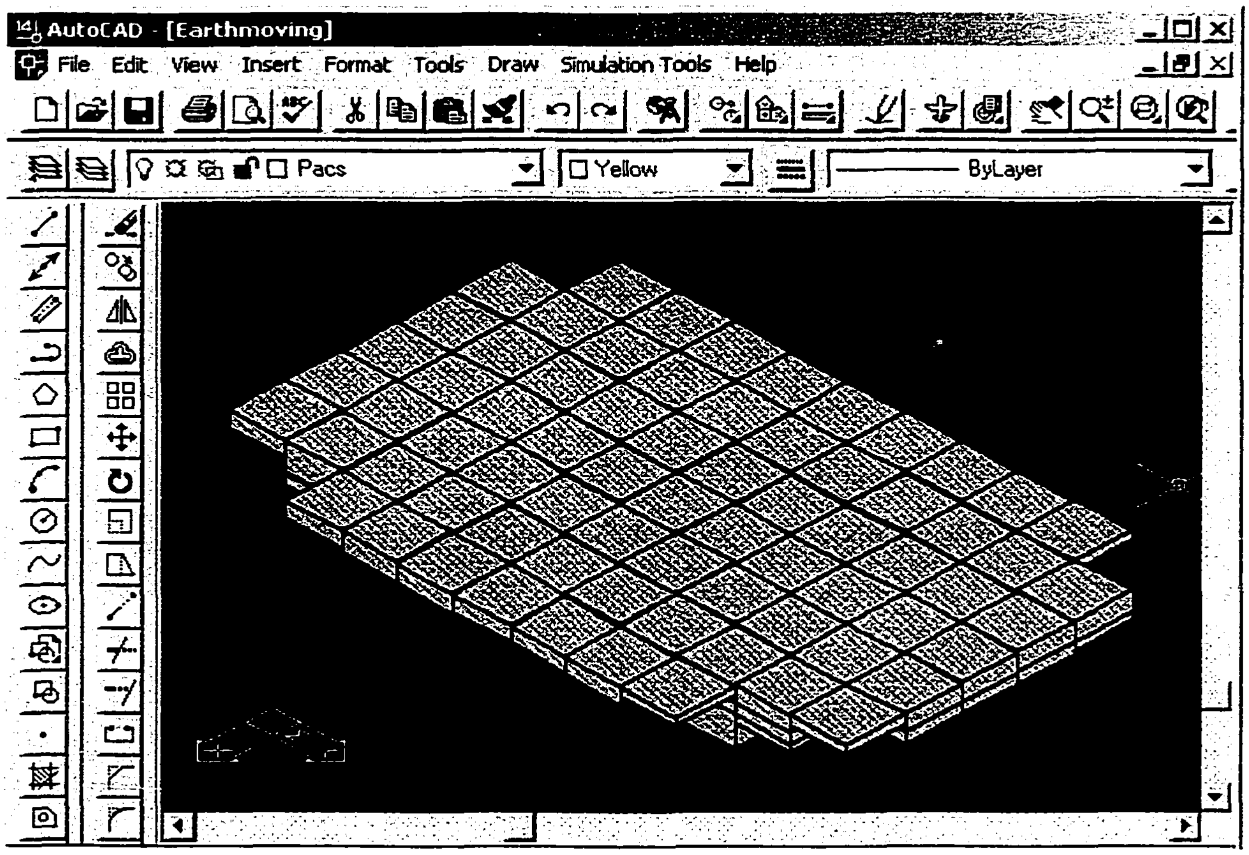


Fig. 6-8 PACs

6.4 Product Network

For common earthmoving operations, two Product Hierarchies must be created to represent source and placement, and three construction phases are involved: preparing, moving, and spreading. Preparations are localized to a given construction source and as such, require one PN to define their sequence. Moving processes involve the source and placement, and as such, require two PNs for their definition. The spreading process is similar to that of preparation. The following steps are taken to create the PNs:

- **Define Construction Phases.** *Add Construction Phase* is used to create a new construction phase that represents a unique construction sequence (Fig. 6-9).

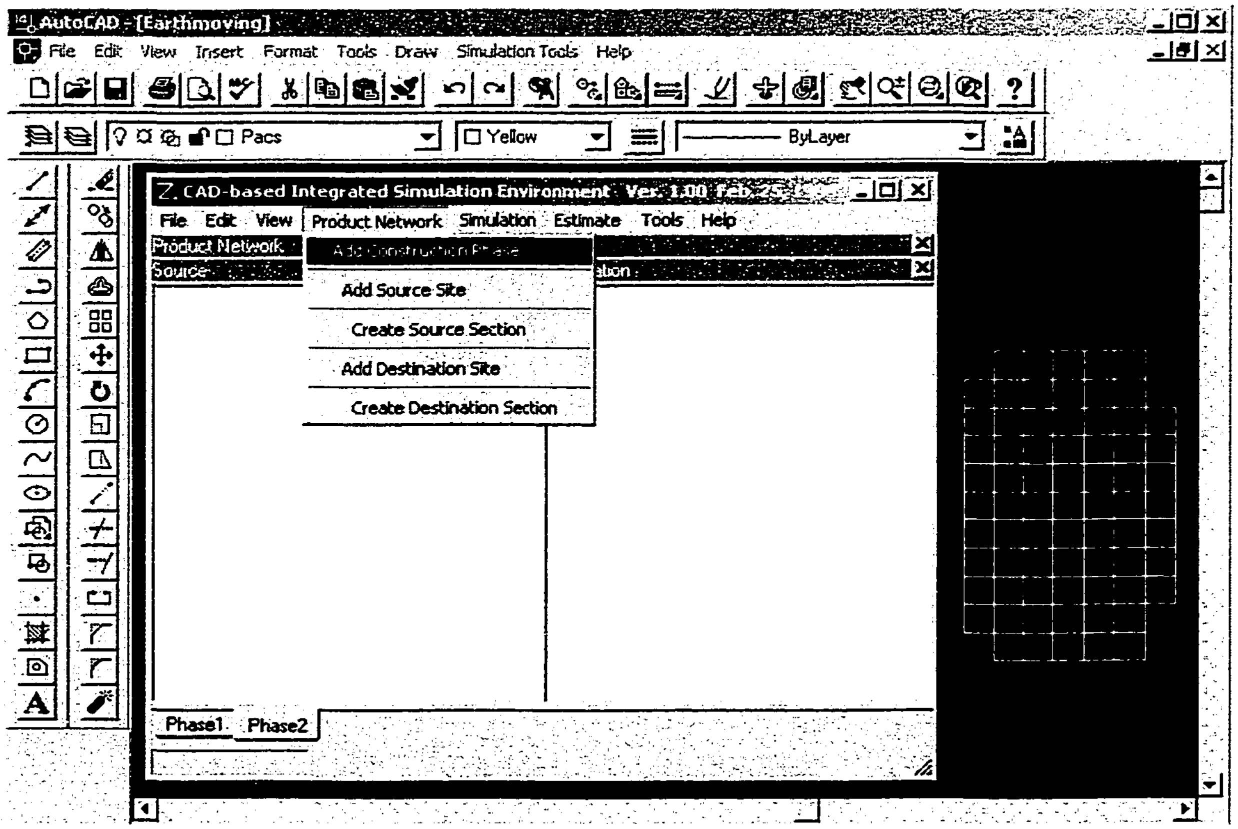

AutoCAD - [Earthmoving]
File Edit View Insert Format Tools Draw Simulation Tools Help
Pacs
Yellow
ByLayer
CAD-based Integrated Simulation Environment Ver. 1.00
File Edit View Product Network Simulation Estimate Tools Help
Product Network
Source
Add Construction Phase
Add Source Site
Create Source Section
Add Destination Site
Create Destination Section
Phase1 Phase2

Fig. 6-9 Add Construction Phase

- **Define Site.** CAD-ISE can manipulate multiple construction sites. *Add Source Site* and *Add Destination Site* are used to add a new site in each construction phase (Fig. 6-10).

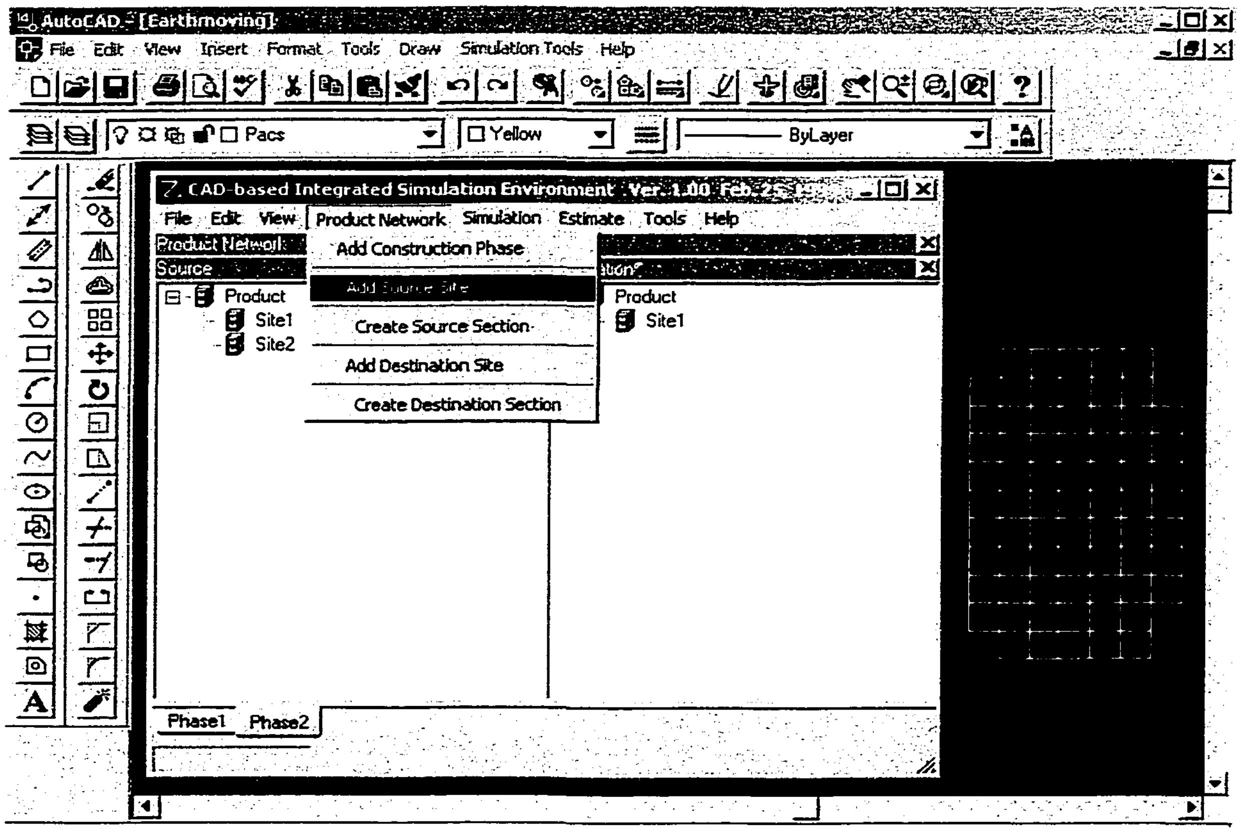

AutoCAD - [Earthmoving]
File Edit View Insert Format Tools Draw Simulation Tools Help
Pacs
Yellow
ByLayer
CAD-based Integrated Simulation Environment Ver. 1.00
File Edit View Product Network Simulation Estimate Tools Help
Product Network
Source
Product
Site1
Site2
Add Construction Phase
Add Source Site
Create Source Section
Add Destination Site
Create Destination Section
Product
Site1
Phase1
Phase2

Fig. 6-10 Add Site

- **Define Section**. *Create Source Section* and *Create Destination Section* are used to specify the number of sections that must be created. CAD-ISE automatically generates the working face(s) and final face, already defined in AutoCAD, by querying the information stored in the AutoCAD file. The sections are then created under each face (Fig. 6-11).

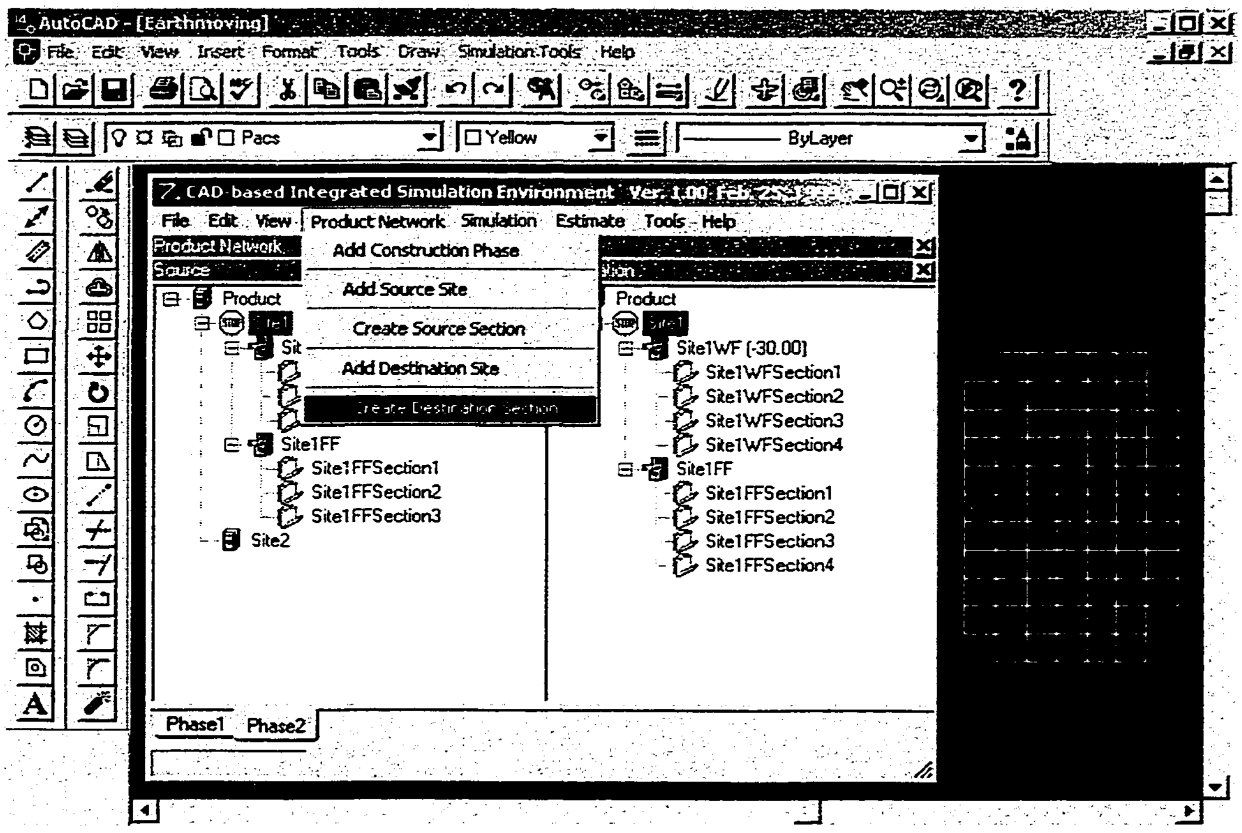


Fig. 6-11 Create Section

- **Define Stake**. The stake is the center of each PAC and every PAC has only one stake. However, each stake usually has more than one PAC pertaining to it. A popup menu item, *Add Stakes,* is used to add stakes for each section. First, the user selects the section to which stakes must be added, then switches to AutoCAD and selects the stakes interactively. This can be done one by one, by selecting each individual stake, or as a group, by specifying the stake's scope. After the stake is created, the PACs that belong to this stake are automatically created as the stake's children. After the PACs are added to the PN, they are hidden to avoid being selected again. Fig. 6-12 illustrates the result of this procedure.

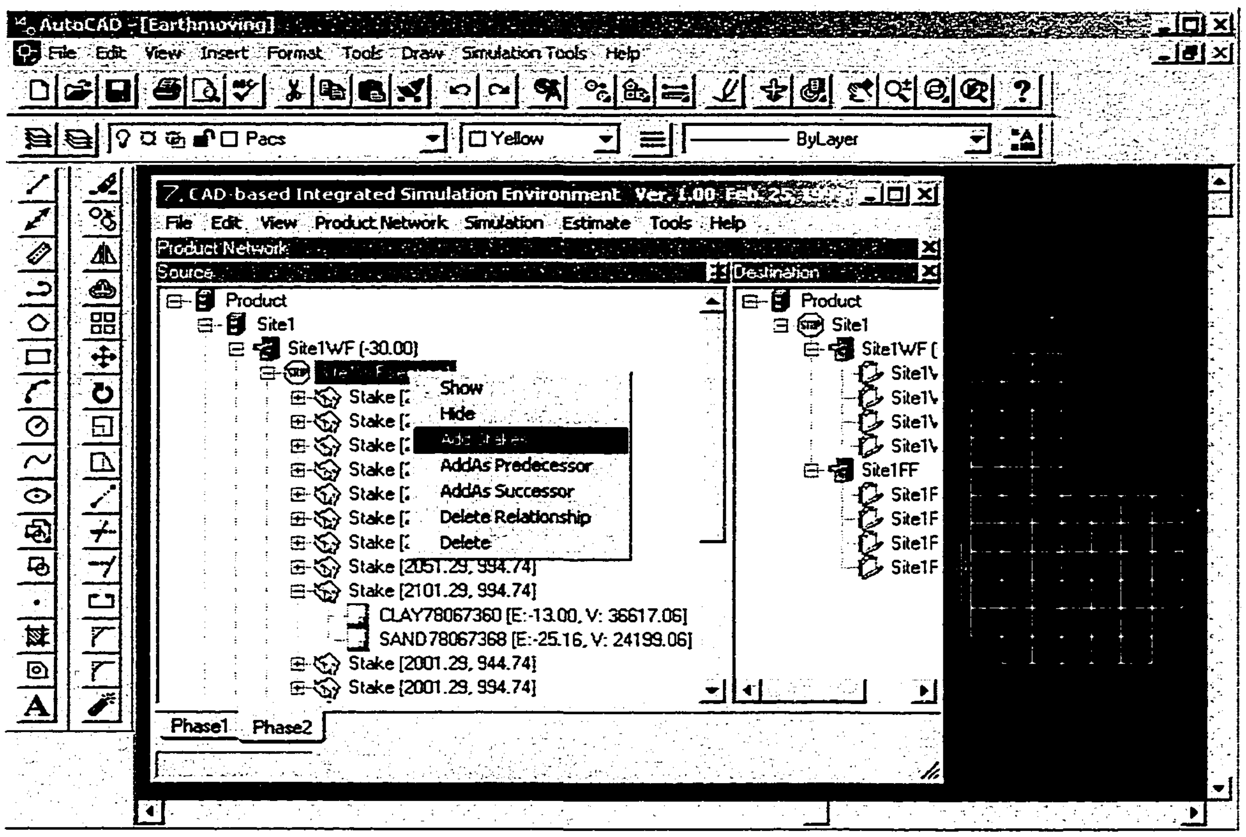

AutoCAD - [Earthmoving]
File Edit View Insert Format Tools Draw Simulation Tools Help
Pacs
Yellow
ByLayer
CAD-based Integrated Simulation Environment Ver. 1.00 Feb.25
File Edit View Product Network Simulation Estimate Tools Help
Product Network
Source
Destination
Product
Site1
Site1WF (-30.00)
Show
Hide
AddAs Predecessor
AddAs Successor
Delete Relationship
Delete
Stake [2101.29, 994.74]
CLAY78067360 [E:-13.00, V: 36617.06]
SAND78067368 [E:-25.16, V: 24199.06]
Stake [2001.29, 944.74]
Stake [2001.29, 994.74]
Site1FF
Phase1
Phase2

Fig. 6-12 Add Stake

- **Define Construction Sequence**. Two popup menu items, *AddAs Predecessor* and *AddAs Successor*, which can be found in all hierarchy levels except the root (Product), are used to define the construction sequence. The user must simply drag and drop one node to another, and specify relationships for predecessors or successors. The relationship of each node can be checked by clicking on the node in question, causing the stop sign icon to appear. If there are any predecessor(s) and/or successor(s) for this node, an arrow icon will appear in the related node(s). The green forward arrow indicates a predecessor and the red backward arrow indicates a successor of this node. If the *Link CAD* option, located in the form popup menu, is checked, the sequence is also shown in

AutoCAD as green, yellow, and red colored PACs, which represent predecessor, current, and successor respectively (Fig. 6-13).

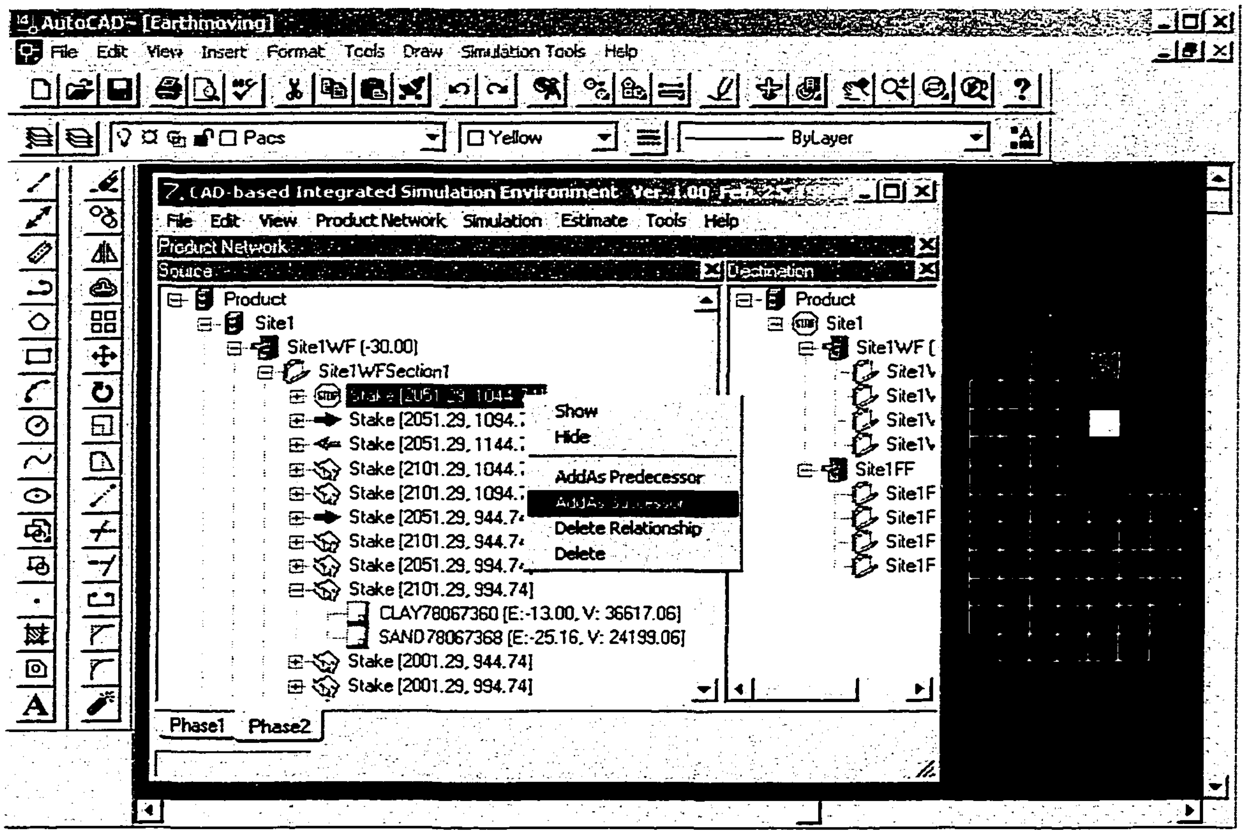


Fig. 6-13 Define Construction Sequence

6.5 Simulation Model

Three special purpose models are built for three different construction phases involved in this earthmoving operation.

The source model is used to represent the preparation activity if the earth is not suitable, and a preparation cycle models the source preparation operation. A dozer created in conjunction with SimPM at run-time works as a product–based entity, and the preparation is an activity in the source model. The placement model is similar to the source model, but the source model refers to the source PH and placement model refers to the placement PH.

The earthmoving model is comprised of excavating, hauling, and dumping. To form a truck cycle, a return activity is added into the model to represent the truck back cycle. In this model, the excavating activity has a sub-model that consists of truck positioning, loading, and leaving to simulate the excavation in more detail.

The resources required for each activity are also defined under the activity sub-model; the procedure used to define the simulation model will be discussed in detail in Chapter 7. The simulation model specifications and implementation reside in Simphony. The CAD_Earthmoving template is created in CAD-ISE, and it will optimize the process of creating a simulation model.

Fig. 6-14 demonstrates the three phases of construction, each phase's resulting PN, and the three corresponding simulation models.

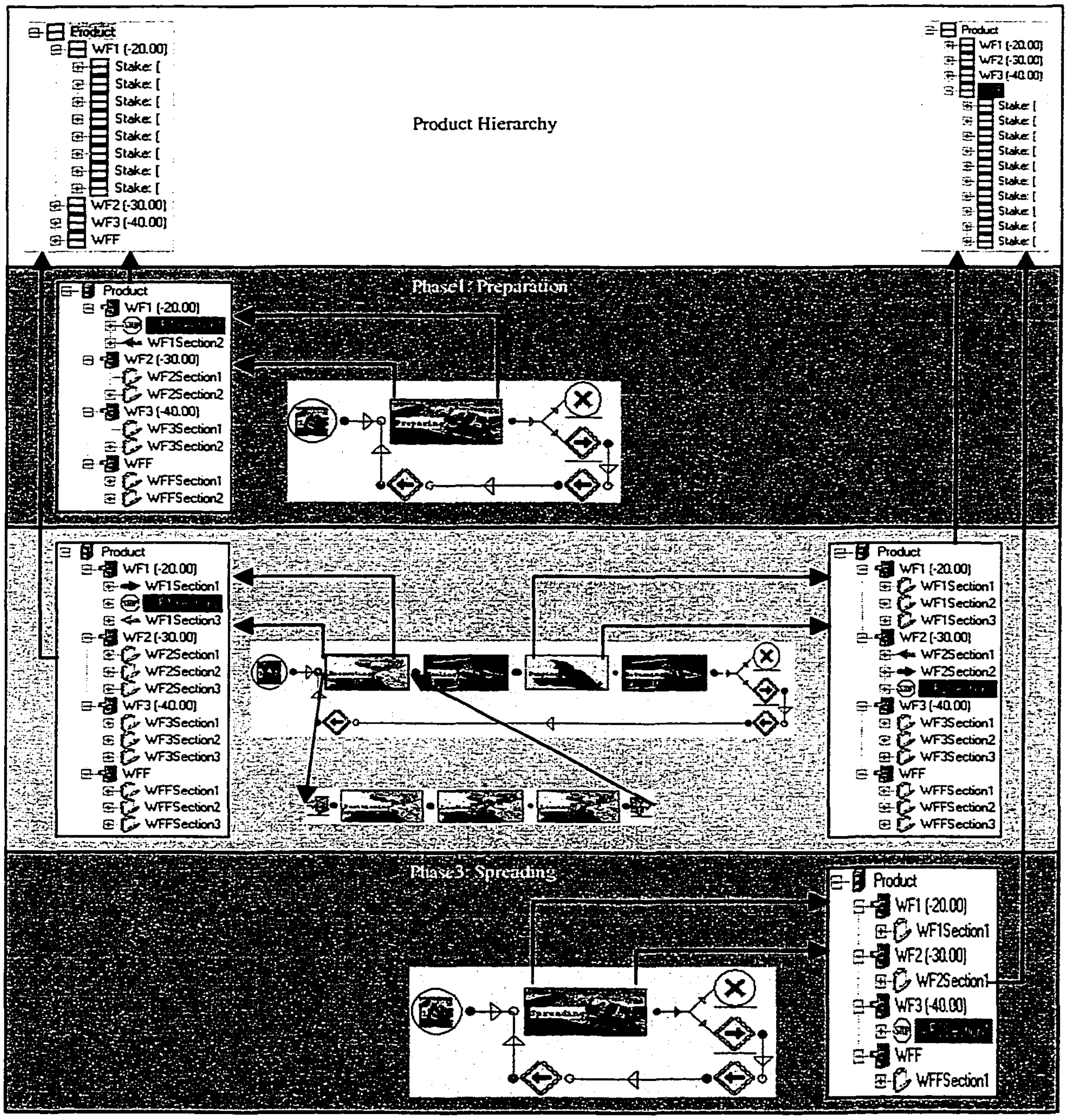


Fig. 6-14 Earthmoving Simulation Architecture

6.6 Product Hierarchy

The Product Hierarchy is generated based on the product network created above, eliminating the section portion. This is done by initializing and linking the simulation model with the Product Networks. Fig. 6-14 shows the generated Product Hierarchies, and Fig. 6-15 shows the dialog box used to accomplish this task.

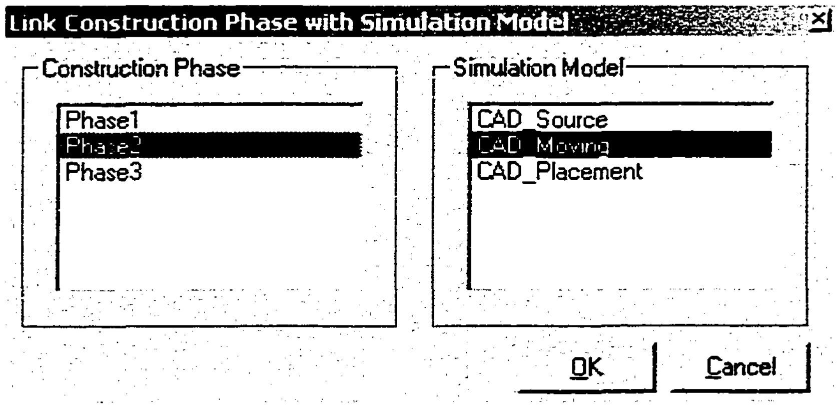
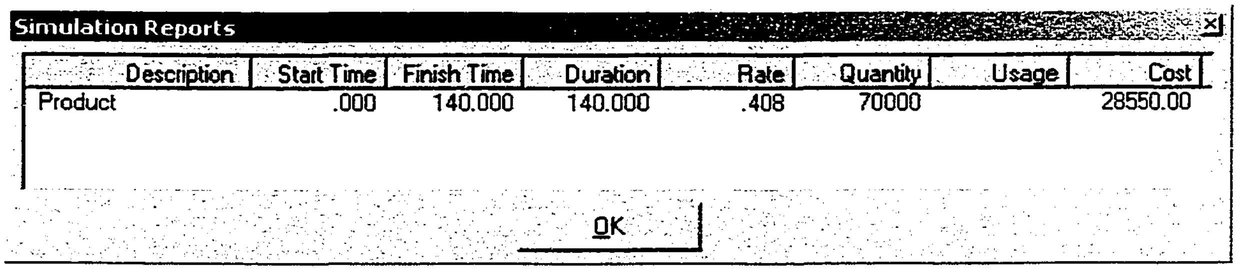


Fig. 6-15 Link Construction Phase with Simulation Model

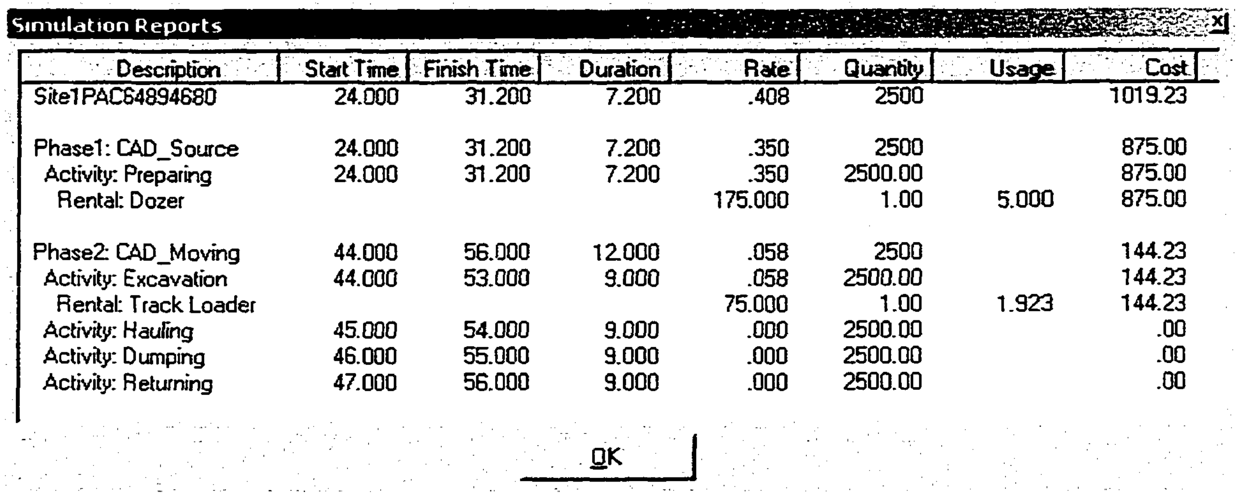
6.7 Simulation Results

After running the simulation, Simulation results are stored in PACs. User can access these results by double clicking any node in the PH and the summary information form (Fig. 6-16) or detail information form (Fig. 6-17) will pop up. The estimate reports can also be generated (Fig. 6-18, 6-19).



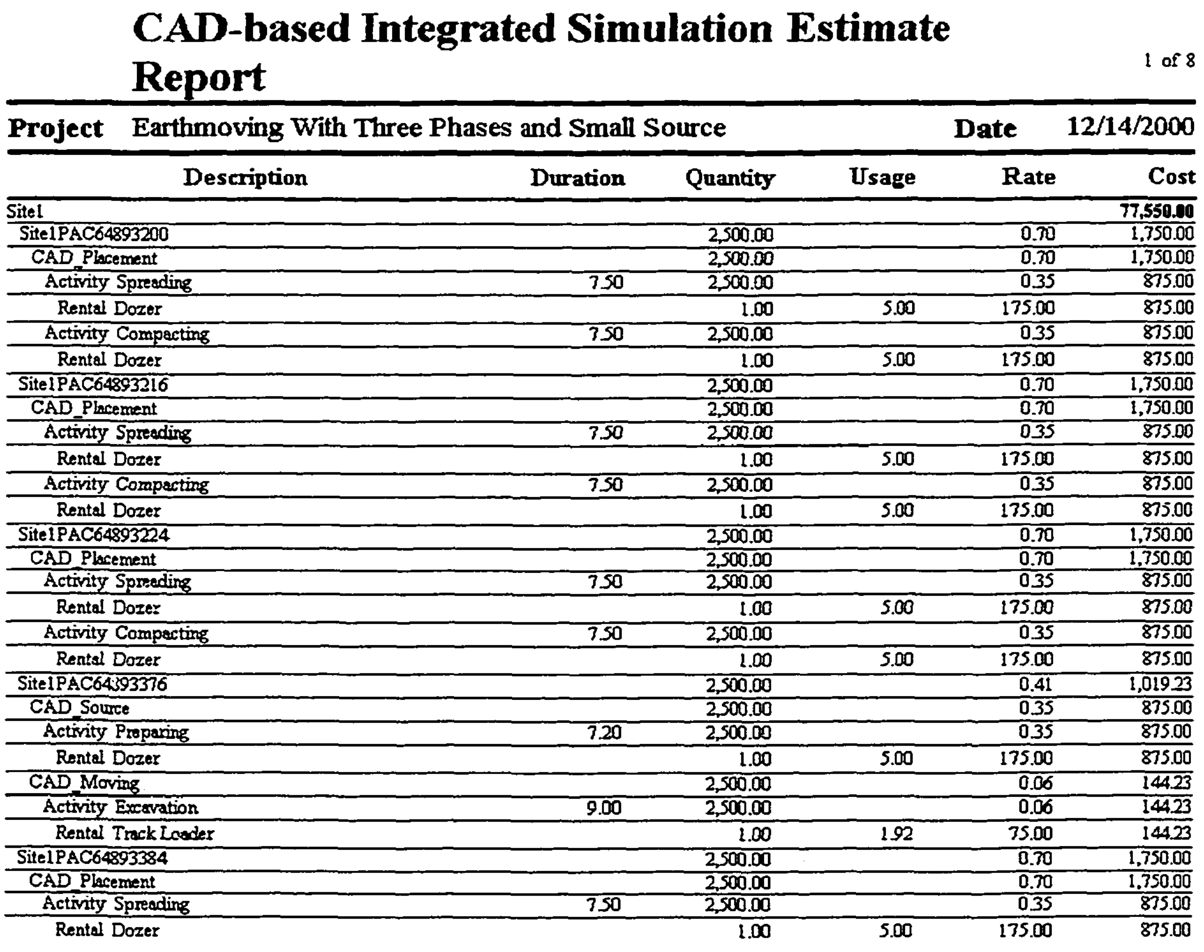
Description	Start Time	Finish Time	Duration	Rate	Quantity	Usage	Cost
Product	.000	140.000	140.000	.408	70000		28550.00

Fig. 6-16 Source Site Duration and Cost



Description	Start Time	Finish Time	Duration	Rate	Quantity	Usage	Cost
Site1PAC64894680	24.000	31.200	7.200	.408	2500		1019.23
Phase1: CAD_Source	24.000	31.200	7.200	.350	2500		875.00
Activity: Preparing	24.000	31.200	7.200	.350	2500.00		875.00
Rental: Dozer				175.000	1.00	5.000	875.00
Phase2: CAD_Moving	44.000	56.000	12.000	.058	2500		144.23
Activity: Excavation	44.000	53.000	9.000	.058	2500.00		144.23
Rental: Track Loader				75.000	1.00	1.923	144.23
Activity: Hauling	45.000	54.000	9.000	.000	2500.00		.00
Activity: Dumping	46.000	55.000	9.000	.000	2500.00		.00
Activity: Returning	47.000	56.000	9.000	.000	2500.00		.00

Fig. 6-17 Source Site Detailed PAC Scheduling and Cost Data



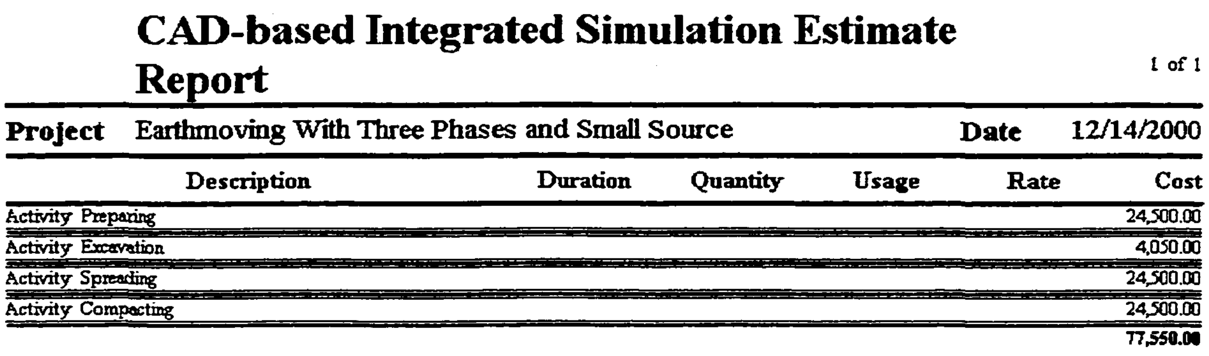
CAD-based Integrated Simulation Estimate Report

1 of 8

Project Earthmoving With Three Phases and Small Source **Date** 12/14/2000

Description	Duration	Quantity	Usage	Rate	Cost
Site1					**77,550.00**
Site1PAC64893200		2,500.00		0.70	1,750.00
CAD_Placement		2,500.00		0.70	1,750.00
Activity Spreading	7.50	2,500.00		0.35	875.00
Rental Dozer		1.00	5.00	175.00	875.00
Activity Compacting	7.50	2,500.00		0.35	875.00
Rental Dozer		1.00	5.00	175.00	875.00
Site1PAC64893216		2,500.00		0.70	1,750.00
CAD_Placement		2,500.00		0.70	1,750.00
Activity Spreading	7.50	2,500.00		0.35	875.00
Rental Dozer		1.00	5.00	175.00	875.00
Activity Compacting	7.50	2,500.00		0.35	875.00
Rental Dozer		1.00	5.00	175.00	875.00
Site1PAC64893224		2,500.00		0.70	1,750.00
CAD_Placement		2,500.00		0.70	1,750.00
Activity Spreading	7.50	2,500.00		0.35	875.00
Rental Dozer		1.00	5.00	175.00	875.00
Activity Compacting	7.50	2,500.00		0.35	875.00
Rental Dozer		1.00	5.00	175.00	875.00
Site1PAC64393376		2,500.00		0.41	1,019.23
CAD_Source		2,500.00		0.35	875.00
Activity Preparing	7.20	2,500.00		0.35	875.00
Rental Dozer		1.00	5.00	175.00	875.00
CAD_Moving		2,500.00		0.06	144.23
Activity Excavation	9.00	2,500.00		0.06	144.23
Rental Track Loader		1.00	1.92	75.00	144.23
Site1PAC64893384		2,500.00		0.70	1,750.00
CAD_Placement		2,500.00		0.70	1,750.00
Activity Spreading	7.50	2,500.00		0.35	875.00
Rental Dozer		1.00	5.00	175.00	875.00

Fig. 6-18 Sample of the Resource Utilization Report



CAD-based Integrated Simulation Estimate Report

1 of 1

Project Earthmoving With Three Phases and Small Source **Date** 12/14/2000

Description	Duration	Quantity	Usage	Rate	Cost
Activity Preparing					24,500.00
Activity Excavation					4,050.00
Activity Spreading					24,500.00
Activity Compacting					24,500.00
					77,550.00

Fig. 6-19 Activity Cost Report

6.8 Conclusions

This chapter walks through the procedure used to build an earthmoving simulation model using CAD-ISE. It presents the concept of product-oriented simulation integrated with CAD, and demonstrates that CAD-ISE provides a simple and straightforward way to build simulation models.

CHAPTER 7 PIPELINE SIMULATION USING CAD-ISE

7.1 Introduction

A linear project consists of a number of similar or identical units that are repetitive in nature. Typical linear projects include pipeline and road construction. This chapter demonstrates CAD-ISE as an alternative simulation tool for linear construction by combining continuous modeling and discrete-event modeling techniques to meet the needs of linear construction simulation. A sample gas line project (J. Shi, S. AbouRizk 1996; J. Shi 1997) is used to illustrate this technique. A comparison of this technique with the traditional technique illustrates the advantages of CAD-ISE.

7.2 Typical Pipeline Construction

Typical pipeline construction could involve the following processes that will be undertaken during the entire period of construction.

The first objective is to undertake a site investigation to ascertain the ground conditions through which the pipeline will be laid in order to finalize the route selected. This is usually carried out by means of a borehole survey with an associated geo-technical analysis.

Once the main pipeline construction begins, the typical sequence of events will be as follows:

1. Right-of-way preparation. The preparations, such as removing obstacles (trees and bushes) and stripping topsoil and subsoil, along the construction route must be done before other construction activities. The productivity for this

event is usually determined by the ground and environment conditions along the line route.

2. Stringing out of pipes. The pipes are delivered to the warehouse in certain lengths before they are transported to the pipeline construction site to be strung out. The productivity of this activity is influenced by geographical and surface conditions on-site.
3. Welding of pipes. Welding is one of the most crucial tasks in pipeline construction. Its productivity is affected by many factors, including welding method, the welder's skill, welding position, and pipe size.
4. Trench excavation. Trench excavation will depend highly on the geo-technical conditions at the excavation site and the resources used for trench excavation. The resources used and the site soil conditions are the most important factors determining productivity.
5. Lowering and laying of pipeline. The pipeline is lowered into the trench using certain equipment. Usually, the crew used to lay the pipe above ground is employed for this task.
6. Back-filling. Care is taken to ensure that no hard material is placed adjacent to the pipeline. Back-filled material is consolidated in certain depth layers.
7. Reinstating of topsoil. Before the topsoil is spread, the subsoil is ripped to a certain depth and any large stones are removed.

7.3 A Sample Pipeline Construction Project

7.3.1 Project Description

A gas line project ten kilometers in length and 30 inches in diameter goes through an area of bush. There are six basic construction processes in the operation: right-of-way, stringing, welding, trenching, lowering-in, and back-filling. The right-of-way is divided into two sections according to the ground conditions. Section 1 is 4 km long with many large trees laying on the route, which makes construction more difficult. Section 2 represents the remaining 6 km, which is covered with bush. The two sections share the same crew. Trench excavation is divided into four sections: section 1 is 3km in length and is excavated by two backhoes, section 2 is a 0.5km valley and is excavated by one backhoe, section 3 is 4km long and is excavated by a trencher, and the remaining 2.5 km is section 4. This final section has conditions similar to section 3, except that some rocks may be encountered, requiring a backhoe to assist the trencher on occasion. Stringing, welding, lowering-in, and back-filling processes have identical performance over the entire route. Welding has its own crew, as does back-filling; stringing and lowering-in share the same crew.

Productivity is defined as the distance (in meters) achieved by a process in one hour. It is determined by the construction site conditions and the resources used in the process. The productivities for this project are obtained from either actual statistics or practitioner's estimates (Table 6-1). Constant (C), Uniform (U), and Triangular (T) distributions are applied to this project. In linear construction, a space buffer is required between two adjacent processes to avoid interference. If the actual leading distance is less than the required buffer, the following process must stop. Table 7-1 summarizes the data information for this project.

Table 7-1 Summary of Project

Process	Section	Section Length (km)	Productivity (m/h)	Resource	Space Buffer (m)
Right-of-way	1	6	C (80)	Crew A	100
	2	4	C (100)	Crew A	
Stringing	1	10	C (400)	Crew B	100
Welding	1	10	U (120,150)	Crew C	100
Trenching	1	3	T (50,70,100)		50
	2	.5	C (50)		
	3	4	C (100)		
	4	2.5	U (80,100)		
Lowering-in	1	10	C (400)	Crew B	20
Back-filling	1	10	C (200)		0

7.3.2 CAD-ISE Pipeline Simulation

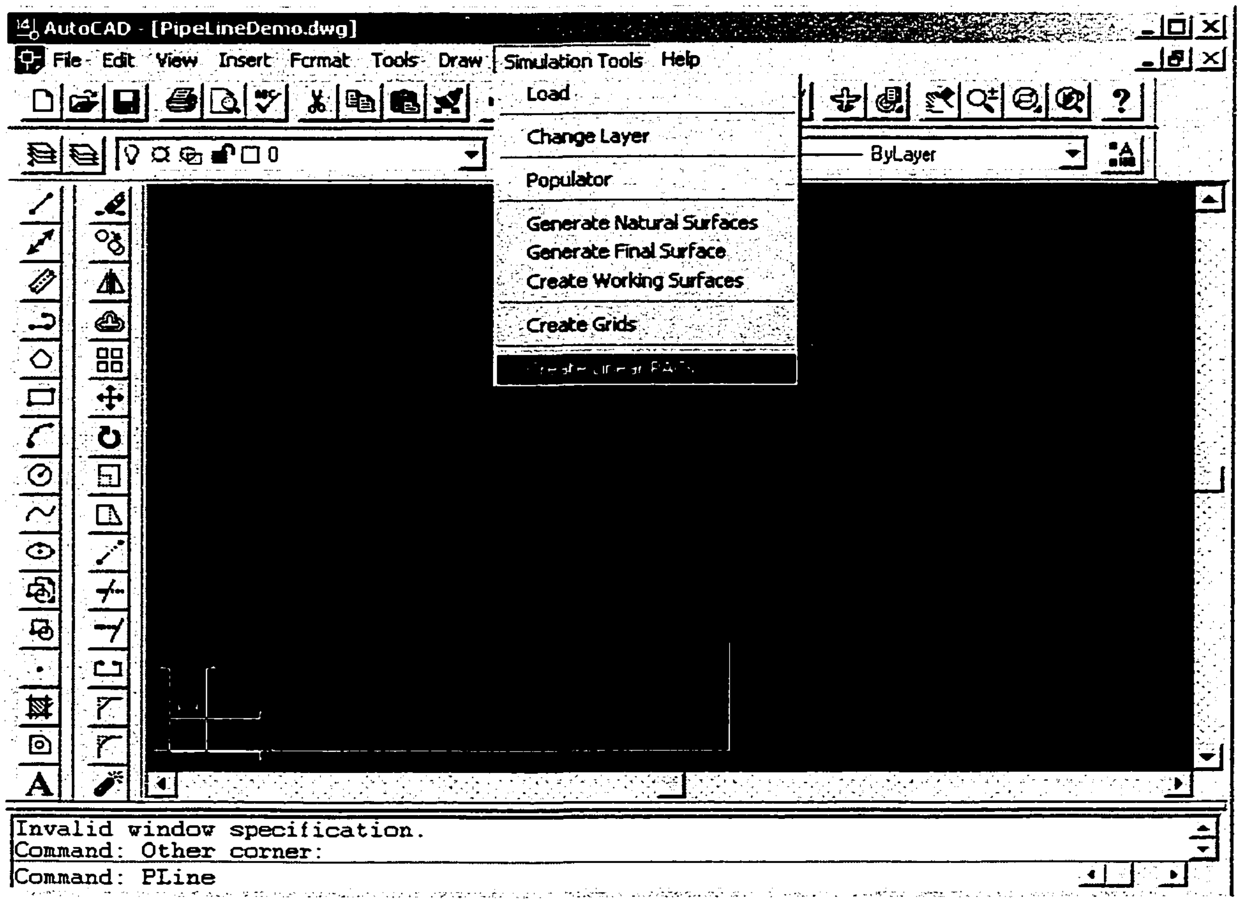
Linear projects are classified into two categories: typical and non-typical repetitive projects. Repetitive activities have identical durations in typical repetitive projects, and are not identical in duration in non-typical repetitive projects. Changes to construction conditions and productivity in a pipeline project during the course of its progress determine its placement into one of the two categories.

A simulation model is classified as either continuous or discrete. In a continuous model, time-dependent variables change continuously over the course of the simulation. In a discrete model, time-dependent variables change discretely at occurred events. A discrete simulation model can be based on event changes and/or process interactions.

7.3.2.1 Product Model – Product Atomic Component (PAC)

In order to experiment with CAD-based simulation, the Product Model – PAC must first be created. Two steps are taken to create the linear Product Model – PAC in CAD:

1. *Defining the pipeline route.* 3D or 2D polylines are used to represent the pipeline route in CAD (Fig. 7-1).

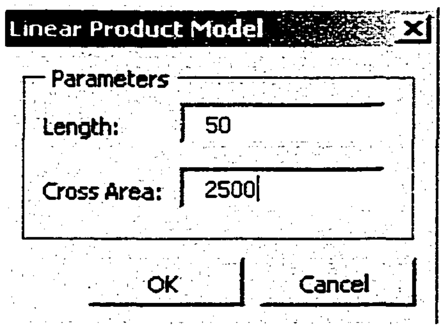

AutoCAD - [PipeLineDemo.dwg]
File Edit View Insert Format Tools Draw Simulation Tools Help
Load
Change Layer
Populator
Generate Natural Surfaces
Generate Final Surface
Create Working Surfaces
Create Grids
ByLayer

```
Invalid window specification.
Command: Other corner:
Command: PLine
```

Fig. 7-1 Pipeline Route Definition in CAD

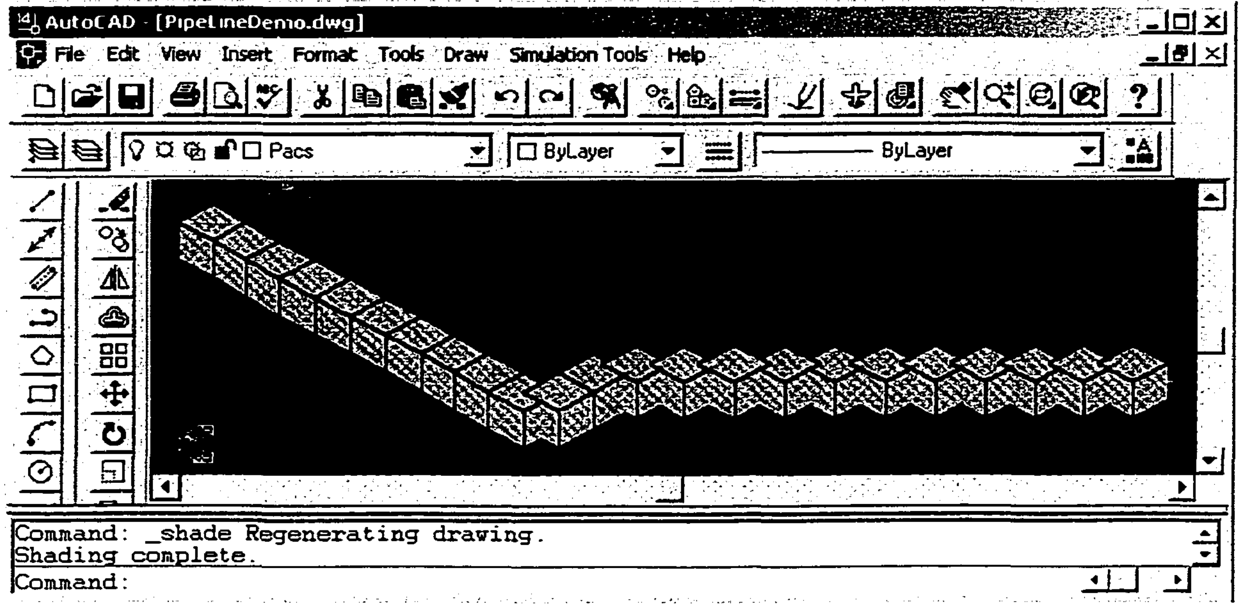
2. *Creating the PACs.* An add-on tool called "Create Linear PACs" is used to create linear PACs (see highlighted menu item in Fig. 7-1). A pop-up form is provided to let the user specify the PAC length, which is usually

determined by the space buffer and the cross-section area (Fig. 7-2). The pipeline PACs are shown in Fig. 7-3.



Length: 50
Cross Area: 2500

Fig. 7-2 Linear PAC Parameters

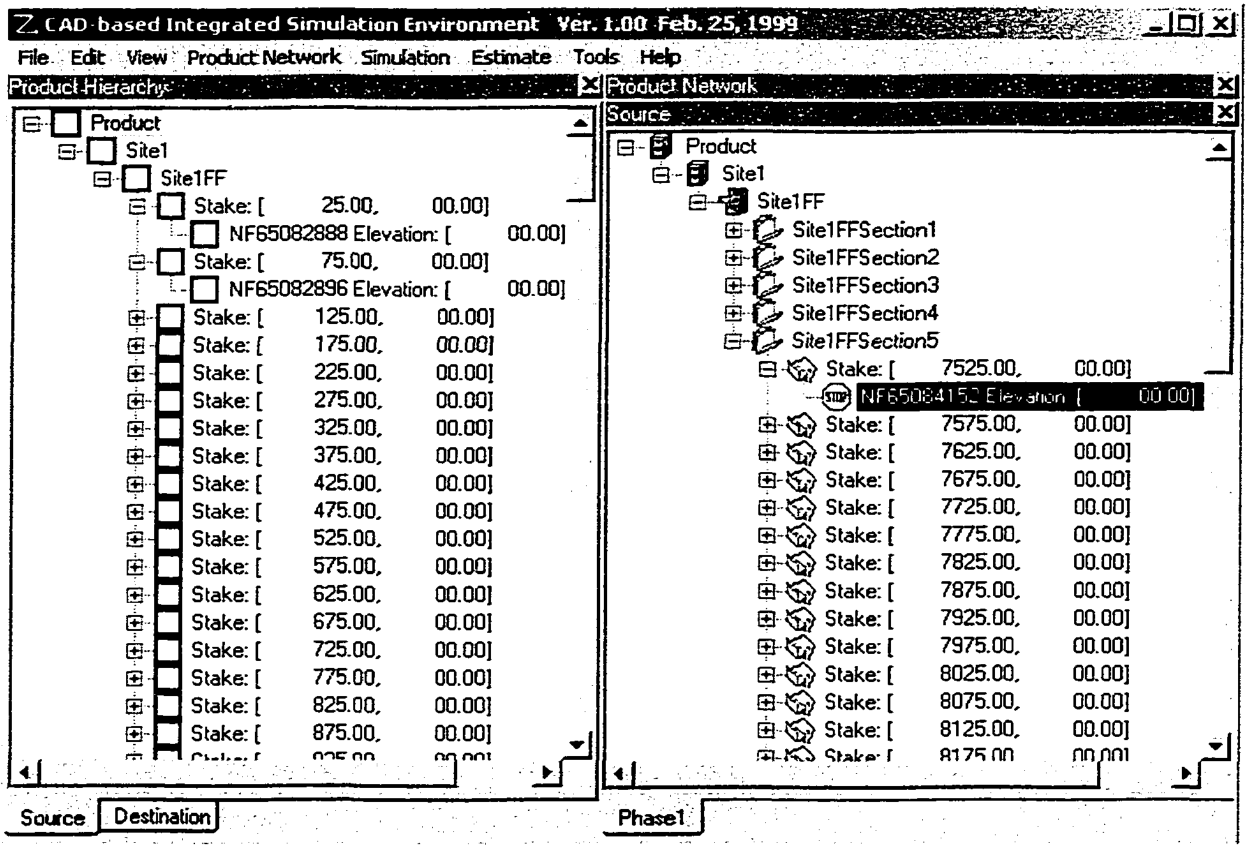

AutoCAD - [PipelineDemo.dwg]
File Edit View Insert Format Tools Draw Simulation Tools Help
Pacs
ByLayer
ByLayer

```
Command: _shade Regenerating drawing.
Shading complete.
Command:
```

Fig. 7-3 PACs of Pipeline (Part)

7.3.2.2 Product Hierarchy and Product Network

For the pipeline project, one Product Hierarchy is created to represent the pipeline product, which assembles all PACs generated in the CAD drawing (Fig. 7-4). All six processes have the same construction sequence, and as such, one PN is defined (Fig. 7-4). In order to share one PN definition, five sections are defined to meet all six processes' identical productivity requirements along the pipeline route. Under the assumption that the start point's stake is 0, the five sections are summarized in Table 7-2.


CAD-based Integrated Simulation Environment Ver. 1.00 Feb. 25, 1999
File Edit View Product Network Simulation Estimate Tools Help
Product Hierarchy
Product Network
Source
Source
Destination
Phase1

```
Product
  Site1
    Site1FF
      Stake: [      25.00,      00.00]
        NF65082888 Elevation: [       00.00]
      Stake: [      75.00,      00.00]
        NF65082896 Elevation: [       00.00]
      Stake: [     125.00,      00.00]
      Stake: [     175.00,      00.00]
      Stake: [     225.00,      00.00]
      Stake: [     275.00,      00.00]
      Stake: [     325.00,      00.00]
      Stake: [     375.00,      00.00]
      Stake: [     425.00,      00.00]
      Stake: [     475.00,      00.00]
      Stake: [     525.00,      00.00]
      Stake: [     575.00,      00.00]
      Stake: [     625.00,      00.00]
      Stake: [     675.00,      00.00]
      Stake: [     725.00,      00.00]
      Stake: [     775.00,      00.00]
      Stake: [     825.00,      00.00]
      Stake: [     875.00,      00.00]
```

```
Product
  Site1
    Site1FF
      Site1FFSection1
      Site1FFSection2
      Site1FFSection3
      Site1FFSection4
      Site1FFSection5
        Stake: [    7525.00,      00.00]
          NF65084152 Elevation: [       00.00]
        Stake: [    7575.00,      00.00]
        Stake: [    7625.00,      00.00]
        Stake: [    7675.00,      00.00]
        Stake: [    7725.00,      00.00]
        Stake: [    7775.00,      00.00]
        Stake: [    7825.00,      00.00]
        Stake: [    7875.00,      00.00]
        Stake: [    7925.00,      00.00]
        Stake: [    7975.00,      00.00]
        Stake: [    8025.00,      00.00]
        Stake: [    8075.00,      00.00]
        Stake: [    8125.00,      00.00]
```

Fig. 7-4 Pipeline PH and PN

Table 7-2 Section Definitions

Section	Range (m)	Process	Productivity (m/h)
1	0 – 3000	Right-of-Way	C (80)
		Stringing	C (400)
		Welding	U (120, 150)
		Trenching	T (50, 70, 100)
		Lowering-in	C (400)
		Back-filling	C (300)
2	3000 – 3500	Right-of-Way	C (80)
		Stringing	C (400)
		Welding	U (120, 150)
		Trenching	C (50)
		Lowering-in	C (400)
		Back-filling	C (300)

3	3500 – 4000	Right-of-Way	C (80)
		Stringing	C (400)
		Welding	U (120, 150)
		Trenching	C (100)
		Lowering-in	C (400)
		Back-filling	C (300)
4	4000 – 7500	Right-of-Way	C (100)
		Stringing	C (400)
		Welding	U (120, 150)
		Trenching	C (100)
		Lowering-in	C (400)
		Back-filling	C (300)
5	7500 – 10000	Right-of-Way	C (100)
		Stringing	C (400)
		Welding	U (120, 150)
		Trenching	U (80, 100)
		Lowering-in	C (400)
		Back-filling	C (300)

7.3.2.3 Simulation Model

Observing the pipeline construction operations from a high level point of view, each repetitive activity is a continuous process. From a conventional Process-Oriented Simulation point of view, six continuous processes can be identified: right-of-way, stringing, welding, trenching, lowering-in, and back-filling, and the status of each process can be defined by the completed distance (e.g. stringing has completed 500m). A continuous simulation model is required to properly suit this situation. However, the presented Product-Oriented Simulation offers an alternative way to model this kind of simulation. The Product-based Entity (PE, refer to Chapters 4 and 5) represents a piece of the product itself. The system status is represented by the PH's status, which is made up of each PAC's status. The six processes are defined to manipulate each PAC, therefore, after finishing any one of the six processes, the PAC's status will change. The change of every PAC's status will cause the system status to change and the system will be updated correspondingly.

The PE can be made small enough to meet the space buffer's requirement and is 10 m long for this example project. Creating a PE is as simple as modifying the CAD_Entity from the CAD modeling element template and specifying the *number of PE* to create, the *Name*, *Product Mode*, and *Capacity*. For this project, only one PE must be created to get the simulation started, and it can be cloned during the course of simulation. *Product mode* is set to *length*, and *product capacity* is 10 m long, which meets the common minimum space buffer requirement.

At the highest level of the pipeline simulation model, there are six activities, one create node, and one destroy node (Fig. 7-5). The product entity is created at the create node, and then passes through six activities – right-of-way, stringing, welding, trenching, lowering-in, and back-filling – before quitting the system by being destroyed by the destroy node. Thus, the simulation model represents the whole pipeline construction cycle in a straightforward manner. The six activities are six identical instances of the CAD_Activity modeling element that are assigned different properties. The following are two detailed examples.

<u>Right-of-way</u>. This activity is the first activity that the PE passes through. A sub-model is used to implement complex situations. Inside the sub-model, there are two activities: *Create Pipe,* which exchanges information with the PH to ensure there is enough pipeline to be further constructed, and *Right-of-way,* which is again further defined by a sub-model. In this sub-model, two parallel activities are defined to address the varying productivity for different sections. The PE that comes from Section 1,2,3 will

be manipulated by activity *Right-of-way 1,2,3* with a productivity of 80 m/h, and PEs from Section 4,5 will be manipulated by activity *Right-of-way 4,5* with a productivity of 100 m/h. Two CAD_filter instances are used to determine to which activity the coming PE should go. A CAD_Resource instance inside *Right-of-way 1,2,3* and *Right-of-way 4,5* is defined to request the resource(s) needed for the operating activity. The CAD_Resource instance is always located at the end of the activity hierarchy.

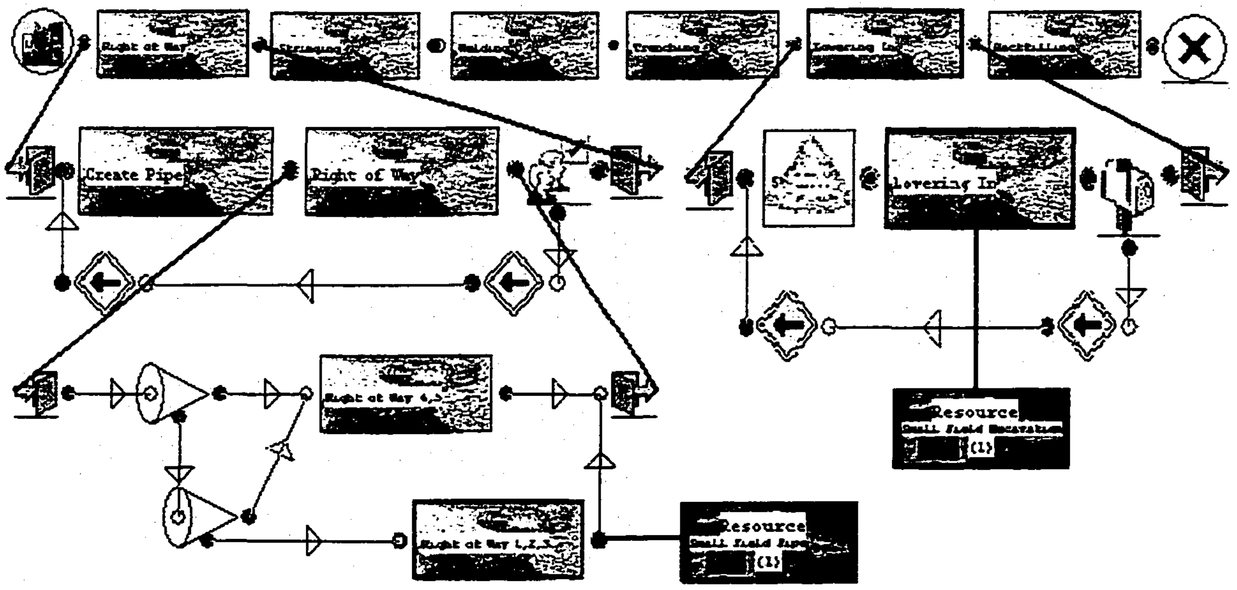


Fig. 7-5 Pipeline Simulation Model

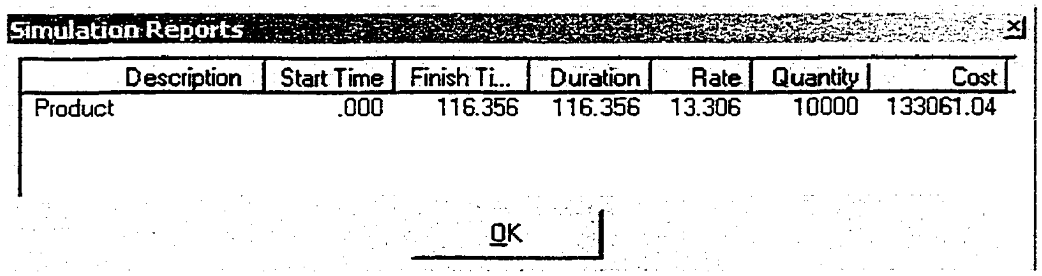
Lowering-in. There is a 50 m leading distance required for the lowering-in process. One CAD_Buffer instance is used in the lowering-in sub-model to accumulate enough PEs (5 * 10 m = 50 m) before starting the lowering-in process. One CAD_Resource instance is also inside the lowering-in activity to request resources.

The trenching process is similar to Right-of-way, but does not have the Create Pipe activity. All other processes, including stringing, welding, and back-filling, are similar to the lowering-in activity.

7.3.2.4 Simulation Results

By experimenting with the model using CAD-ISE, it is determined that the project duration is 116.4 with a total cost of $133,061.04 (Fig. 7-6). Detailed PAC scheduling and cost information can also be obtained (Fig. 7-7). Fig. 7-8 gives a detailed resource utilization report in terms of PACs.

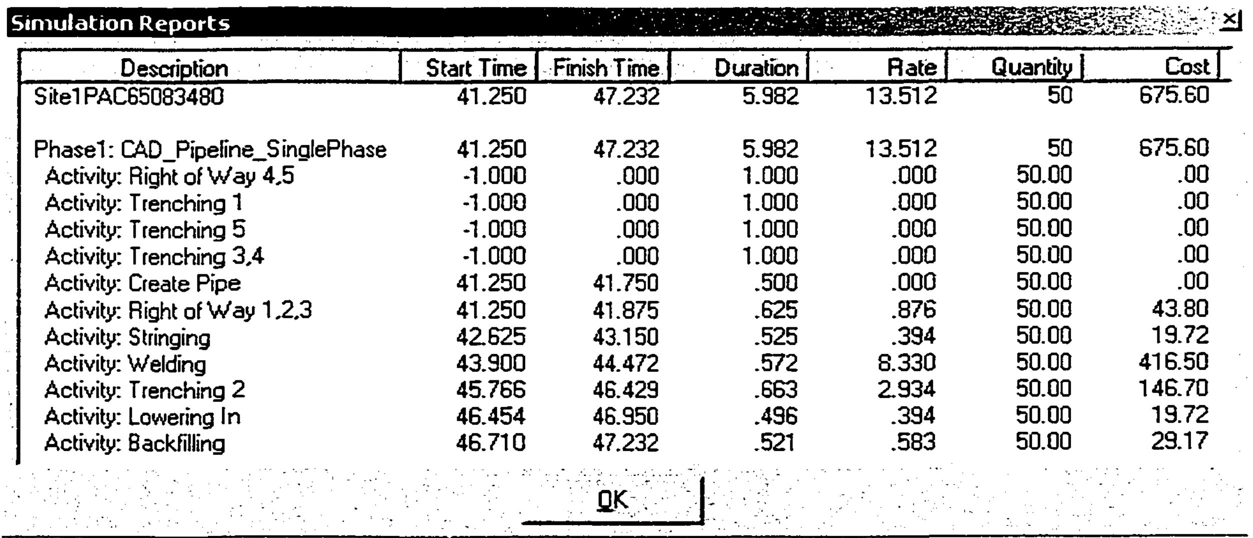
As the duration determined in the sample project literature is 115 (J. Shi, S. AbouRizk 1996; J. Shi 1997), the duration 116.4 generated from CAD-ISE is satisfactory.



Simulation Reports

Description	Start Time	Finish Ti...	Duration	Rate	Quantity	Cost
Product	.000	116.356	116.356	13.306	10000	133061.04

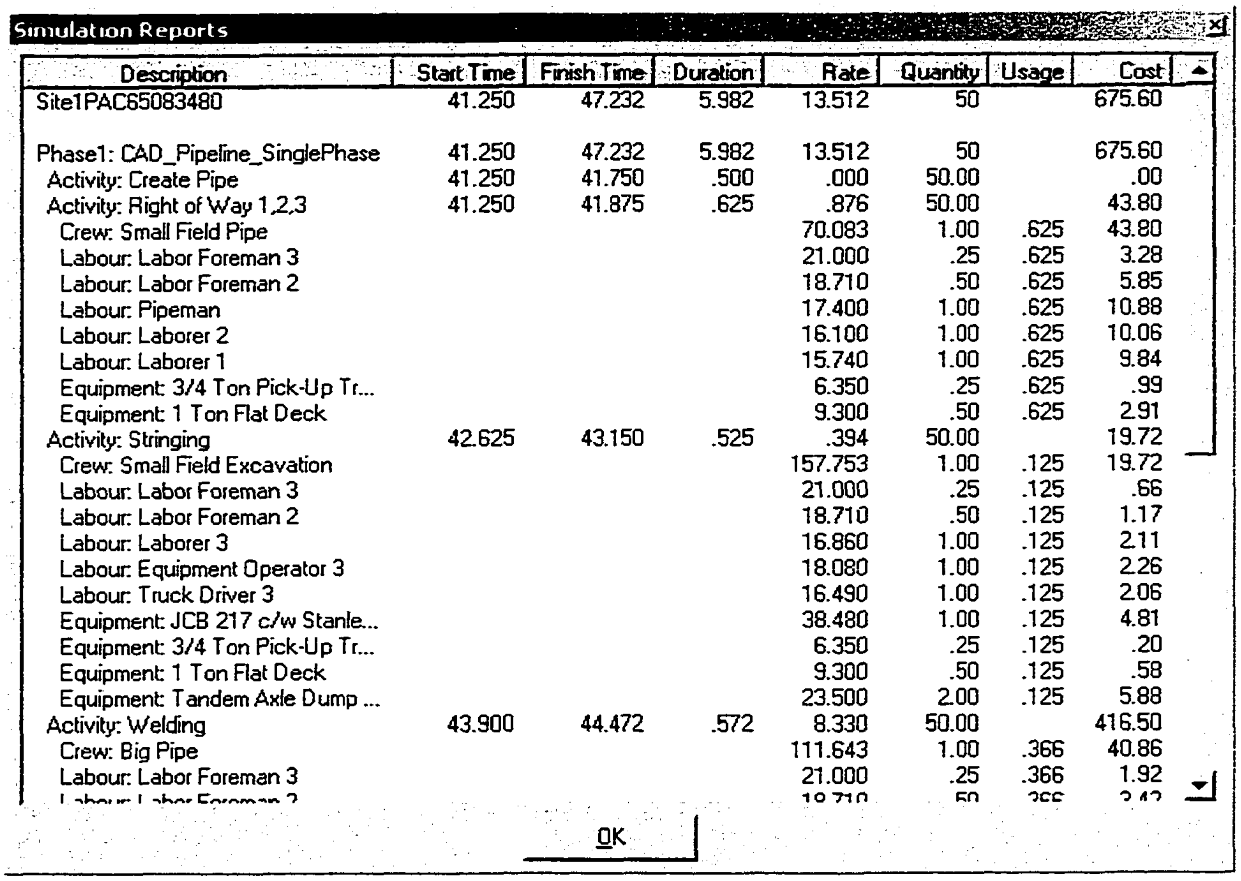
Fig. 7-6 Project Duration and Total Cost



Simulation Reports

Description	Start Time	Finish Time	Duration	Rate	Quantity	Cost
Site1PAC65083480	41.250	47.232	5.982	13.512	50	675.60
Phase1: CAD_Pipeline_SinglePhase	41.250	47.232	5.982	13.512	50	675.60
Activity: Right of Way 4,5	-1.000	.000	1.000	.000	50.00	.00
Activity: Trenching 1	-1.000	.000	1.000	.000	50.00	.00
Activity: Trenching 5	-1.000	.000	1.000	.000	50.00	.00
Activity: Trenching 3,4	-1.000	.000	1.000	.000	50.00	.00
Activity: Create Pipe	41.250	41.750	.500	.000	50.00	.00
Activity: Right of Way 1,2,3	41.250	41.875	.625	.876	50.00	43.80
Activity: Stringing	42.625	43.150	.525	.394	50.00	19.72
Activity: Welding	43.900	44.472	.572	8.330	50.00	416.50
Activity: Trenching 2	45.766	46.429	.663	2.934	50.00	146.70
Activity: Lowering In	46.454	46.950	.496	.394	50.00	19.72
Activity: Backfilling	46.710	47.232	.521	.583	50.00	29.17

Fig. 7-7 Detailed PAC Scheduling and Cost Data



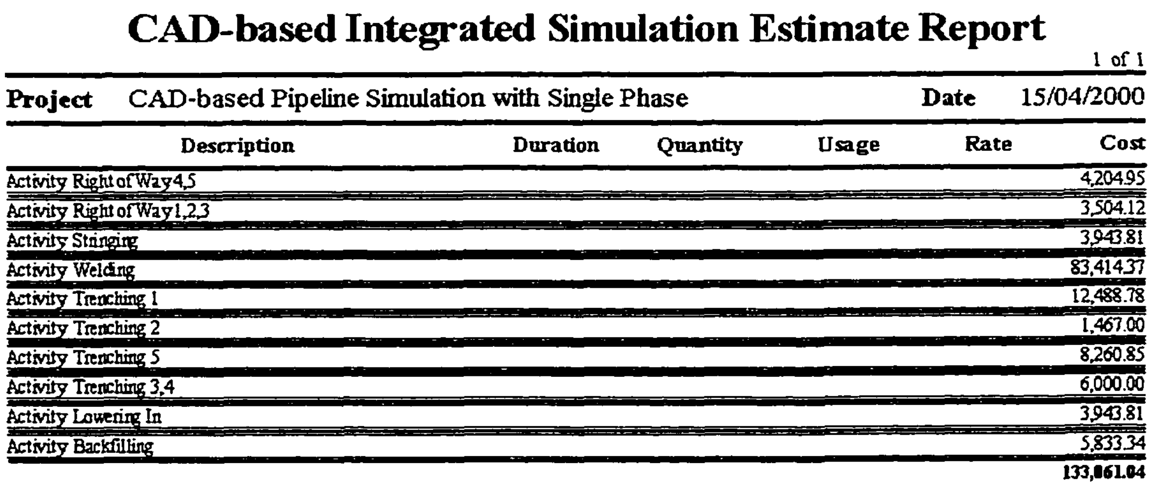
Simulation Reports

Description	Start Time	Finish Time	Duration	Rate	Quantity	Usage	Cost
Site1PAC65083480	41.250	47.232	5.982	13.512	50		675.60
Phase1: CAD_Pipeline_SinglePhase	41.250	47.232	5.982	13.512	50		675.60
Activity: Create Pipe	41.250	41.750	.500	.000	50.00		.00
Activity: Right of Way 1,2,3	41.250	41.875	.625	.876	50.00		43.80
Crew: Small Field Pipe				70.083	1.00	.625	43.80
Labour: Labor Foreman 3				21.000	.25	.625	3.28
Labour: Labor Foreman 2				18.710	.50	.625	5.85
Labour: Pipeman				17.400	1.00	.625	10.88
Labour: Laborer 2				16.100	1.00	.625	10.06
Labour: Laborer 1				15.740	1.00	.625	9.84
Equipment: 3/4 Ton Pick-Up Tr...				6.350	.25	.625	.99
Equipment: 1 Ton Flat Deck				9.300	.50	.625	2.91
Activity: Stringing	42.625	43.150	.525	.394	50.00		19.72
Crew: Small Field Excavation				157.753	1.00	.125	19.72
Labour: Labor Foreman 3				21.000	.25	.125	.66
Labour: Labor Foreman 2				18.710	.50	.125	1.17
Labour: Laborer 3				16.860	1.00	.125	2.11
Labour: Equipment Operator 3				18.080	1.00	.125	2.26
Labour: Truck Driver 3				16.490	1.00	.125	2.06
Equipment: JCB 217 c/w Stanle...				38.480	1.00	.125	4.81
Equipment: 3/4 Ton Pick-Up Tr...				6.350	.25	.125	.20
Equipment: 1 Ton Flat Deck				9.300	.50	.125	.58
Equipment: Tandem Axle Dump ...				23.500	2.00	.125	5.88
Activity: Welding	43.900	44.472	.572	8.330	50.00		416.50
Crew: Big Pipe				111.643	1.00	.366	40.86
Labour: Labor Foreman 3				21.000	.25	.366	1.92
Labour: Labor Foreman 2				18.710	.50	.366	2.42

OK

Fig. 7-8 Detailed Resource Utilization

After the simulation completes, the estimate and scheduling data and detailed reports are generated (Fig. 7-9, 7-10, 7-11) using functionality within CAD-ISE.



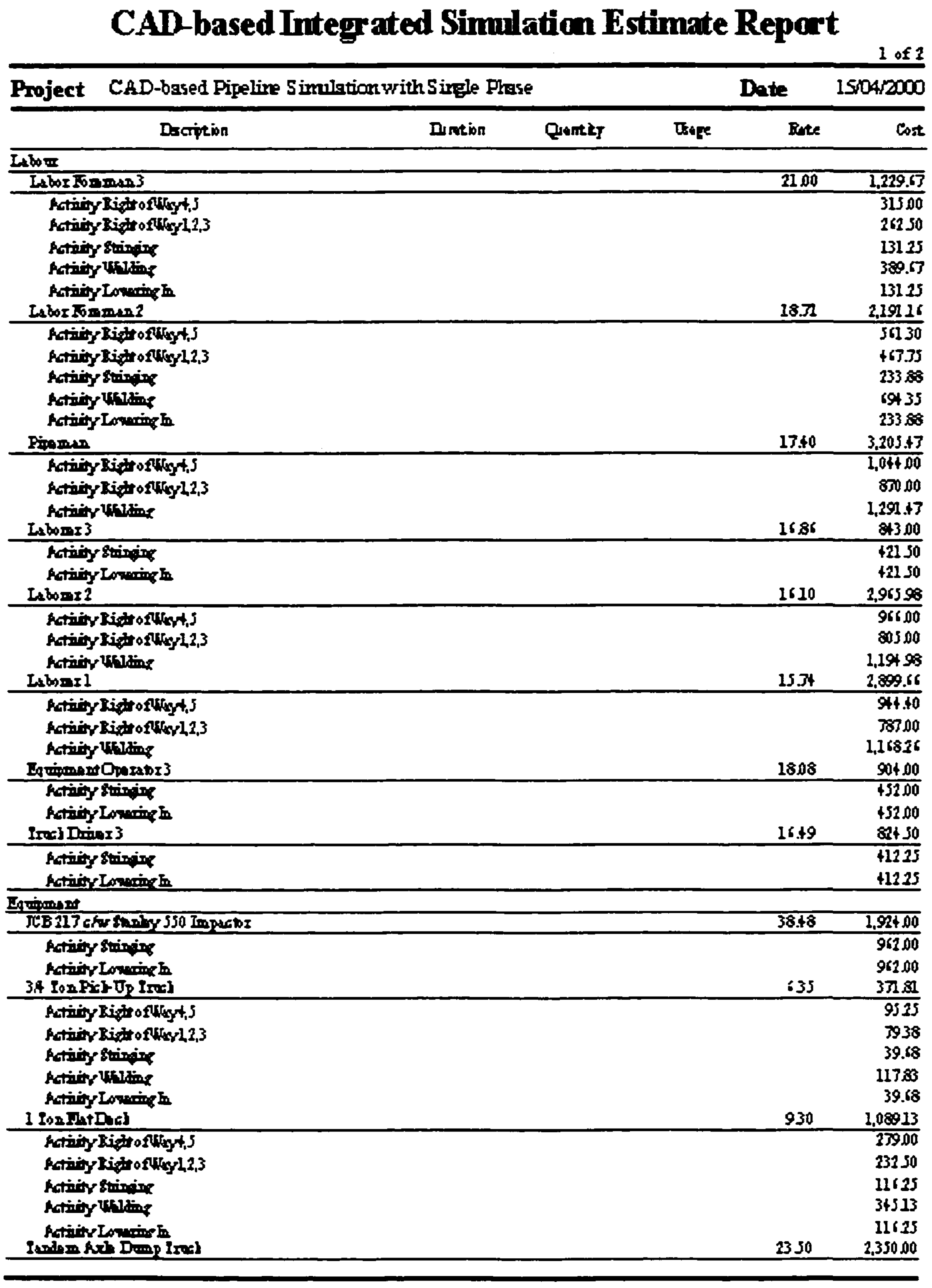
CAD-based Integrated Simulation Estimate Report

1 of 1

Project CAD-based Pipeline Simulation with Single Phase **Date** 15/04/2000

Description	Duration	Quantity	Usage	Rate	Cost
Activity Right of Way4,5					4,204.95
Activity Right of Way1,2,3					3,504.12
Activity Stringing					3,943.81
Activity Welding					83,414.37
Activity Trenching 1					12,488.78
Activity Trenching 2					1,467.00
Activity Trenching 5					8,260.85
Activity Trenching 3,4					6,000.00
Activity Lowering In					3,943.81
Activity Backfilling					5,833.34
					133,061.04

Fig. 7-9 Activity Cost Report



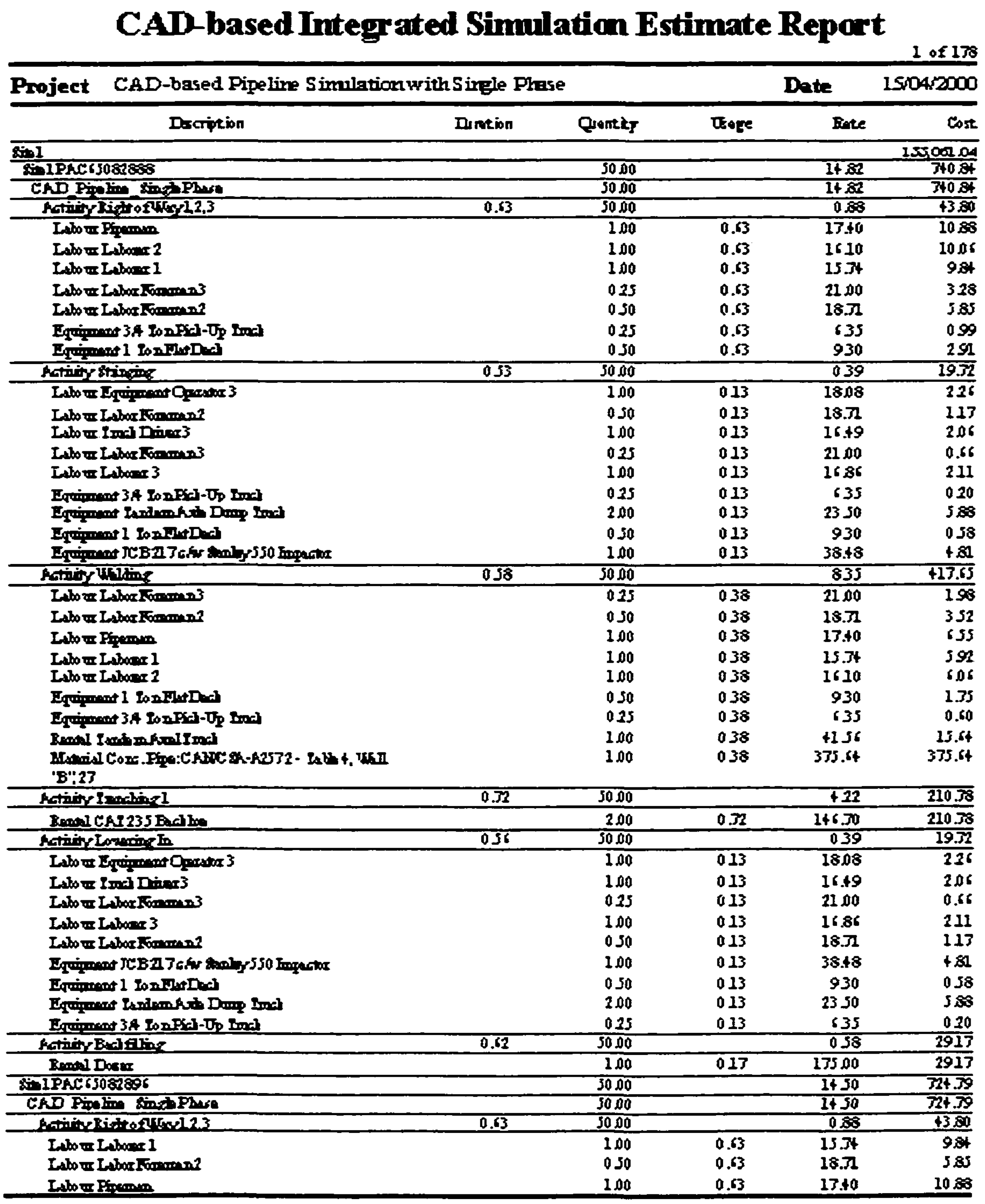
CAD-based Integrated Simulation Estimate Report

1 of 2

Project CAD-based Pipeline Simulation with Single Phase **Date** 15/04/2000

Description	Duration	Quantity	Usage	Rate	Cost
Labour					
Labor Foreman 3				21.00	1,229.67
Activity Right of Way 4,5					315.00
Activity Right of Way 1,2,3					262.50
Activity Stringing					131.25
Activity Welding					389.67
Activity Lowering In					131.25
Labor Foreman 2				18.71	2,191.16
Activity Right of Way 4,5					561.30
Activity Right of Way 1,2,3					467.75
Activity Stringing					233.88
Activity Welding					694.35
Activity Lowering In					233.88
Pipeman				17.40	3,205.47
Activity Right of Way 4,5					1,044.00
Activity Right of Way 1,2,3					870.00
Activity Welding					1,291.47
Labourer 3				16.86	843.00
Activity Stringing					421.50
Activity Lowering In					421.50
Labourer 2				16.10	2,965.98
Activity Right of Way 4,5					966.00
Activity Right of Way 1,2,3					805.00
Activity Welding					1,194.98
Labourer 1				15.74	2,899.66
Activity Right of Way 4,5					944.40
Activity Right of Way 1,2,3					787.00
Activity Welding					1,168.26
Equipment Operator 3				18.08	904.00
Activity Stringing					452.00
Activity Lowering In					452.00
Truck Driver 3				16.49	824.50
Activity Stringing					412.25
Activity Lowering In					412.25
Equipment					
JCB 217 c/w Stanley 550 Impactor				38.48	1,924.00
Activity Stringing					962.00
Activity Lowering In					962.00
3/4 Ton Pick-Up Truck				6.35	371.81
Activity Right of Way 4,5					95.25
Activity Right of Way 1,2,3					79.38
Activity Stringing					39.68
Activity Welding					117.83
Activity Lowering In					39.68
1 Ton Flat Deck				9.30	1,089.13
Activity Right of Way 4,5					279.00
Activity Right of Way 1,2,3					232.50
Activity Stringing					116.25
Activity Welding					345.13
Activity Lowering In					116.25
Tandem Axle Dump Truck				23.50	2,350.00

Fig. 7-10 Resource Summary Report



CAD-based Integrated Simulation Estimate Report

1 of 178

Project CAD-based Pipeline Simulation with Single Phase **Date** 15/04/2000

Description	Duration	Quantity	Usage	Rate	Cost
Sim1					153,061.04
Sim1PAC65082888		50.00		14.82	740.84
CAD_Pipeline_SinglePhase		50.00		14.82	740.84
Activity Right of Way1,2,3	0.63	50.00		0.88	43.80
Labour Pipeman		1.00	0.63	17.40	10.88
Labour Labourer 2		1.00	0.63	16.10	10.06
Labour Labourer 1		1.00	0.63	15.74	9.84
Labour Labor Foreman3		0.25	0.63	21.00	3.28
Labour Labor Foreman2		0.50	0.63	18.71	5.85
Equipment 3/4 Ton Pick-Up Truck		0.25	0.63	6.35	0.99
Equipment 1 Ton Flat Deck		0.50	0.63	9.30	2.91
Activity Stringing	0.53	50.00		0.39	19.72
Labour Equipment Operator 3		1.00	0.13	18.08	2.26
Labour Labor Foreman2		0.50	0.13	18.71	1.17
Labour Truck Driver3		1.00	0.13	16.49	2.06
Labour Labor Foreman3		0.25	0.13	21.00	0.66
Labour Labourer 3		1.00	0.13	16.86	2.11
Equipment 3/4 Ton Pick-Up Truck		0.25	0.13	6.35	0.20
Equipment Tandem Axle Dump Truck		2.00	0.13	23.50	5.88
Equipment 1 Ton Flat Deck		0.50	0.13	9.30	0.58
Equipment JCB217c/w Stanley550 Impactor		1.00	0.13	38.48	4.81
Activity Welding	0.58	50.00		8.35	417.65
Labour Labor Foreman3		0.25	0.38	21.00	1.98
Labour Labor Foreman2		0.50	0.38	18.71	3.52
Labour Pipeman		1.00	0.38	17.40	6.55
Labour Labourer 1		1.00	0.38	15.74	5.92
Labour Labourer 2		1.00	0.38	16.10	6.06
Equipment 1 Ton Flat Deck		0.50	0.38	9.30	1.75
Equipment 3/4 Ton Pick-Up Truck		0.25	0.38	6.35	0.60
Rental Tandem Axial Truck		1.00	0.38	41.56	15.64
Material Conc. Pipe:CANC SA-A2572 - Table 4, Wall "B";27		1.00	0.38	375.64	375.64
Activity Trenching1	0.72	50.00		4.22	210.78
Rental CAT 235 Backhoe		2.00	0.72	146.70	210.78
Activity Lowering In	0.56	50.00		0.39	19.72
Labour Equipment Operator 3		1.00	0.13	18.08	2.26
Labour Truck Driver3		1.00	0.13	16.49	2.06
Labour Labor Foreman3		0.25	0.13	21.00	0.66
Labour Labourer 3		1.00	0.13	16.86	2.11
Labour Labor Foreman2		0.50	0.13	18.71	1.17
Equipment JCB217c/w Stanley550 Impactor		1.00	0.13	38.48	4.81
Equipment 1 Ton Flat Deck		0.50	0.13	9.30	0.58
Equipment Tandem Axle Dump Truck		2.00	0.13	23.50	5.88
Equipment 3/4 Ton Pick-Up Truck		0.25	0.13	6.35	0.20
Activity Backfilling	0.62	50.00		0.58	29.17
Rental Dozer		1.00	0.17	175.00	29.17
Sim1PAC65082896		50.00		14.50	724.79
CAD_Pipeline_SinglePhase		50.00		14.50	724.79
Activity Right of Way1,2,3	0.63	50.00		0.88	43.80
Labour Labourer 1		1.00	0.63	15.74	9.84
Labour Labor Foreman2		0.50	0.63	18.71	5.85
Labour Pipeman		1.00	0.63	17.40	10.88

Fig. 7-11 Product Scheduling, Cost, and Resource Utilization Report

7.4 Conclusion

The pipeline simulation model developed under CAD-ISE in this chapter demonstrates the advantages of product-oriented simulation methodology applied to a repetitive construction project. It provides an alternate way to conduct continuous simulation by employing Product-based Entities (PE) that advance the system's status through the updating of each PAC's status. The PE can be made small enough to suit the continuous model's requirement. Additionally, CAD-ISE is built using a discrete simulation engine, which offers all of the advantages of a discrete simulation, including detailed modeling of resource operation.

Because a process is treated as a continuous process until it is complete, it would be very difficult to model resource sharing in a continuous model, which can only be realized through user-inserted subroutines. In a CAD-ISE model, the start of a resource cycle is an event. Resources are seized at that event and are freed after completing an operation cycle. Resource sharing can be easily achieved by creating an instance of the CAD_Resource modeling element.

CAD-ISE is flexible enough to model complex construction conditions without any coding, including different construction sequences and multiple starting points for each process, by using multi-construction phases.

CHAPTER 8 CONCLUSIONS

8.1 Summary of Research

This thesis presented a CAD-based integrated simulation environment that applies a product-oriented simulation methodology for simulating construction projects. The primary objective behind CAD-ISE is to integrate a CAD system with a simulation system and use the information in the CAD system, which was previously created in the design stage, to prepare the product model for simulation and define a simulation model from the pre-defined template in the same environment. Four secondary goals are accomplished in achieving the primary objective:

1. To develop the framework of a CAD-based Integrated Simulation Environment. Chapter 1 and Chapter 2 described the entire CAD-based Integrated Simulation Environment and the background of the research, defined its scope, and identified the problems to be solved.
2. To conceptualize the product-oriented simulation methodology. Chapter 3 detailed the conceptual design of the product-oriented simulation methodology used by CAD-ISE. The methodology provides an alternative way to design a simulation system by changing its perspective. The benefits of this methodology were also explained.
3. To implement the prototype system based on the framework and methodology presented in Chapters 1, 2, and 3. Chapter 4 described the actual development of the prototype system, and provided the discussions

of the object class and system features. The modeling elements developed to conduct CAD-based simulation were also documented in Chapters 4 and 5. The developed prototype system proved the framework and the methodology of the research.

4. To conduct a case study. Chapters 6 and 7 provided two case studies to test and evaluate the prototype system. These two case studies demonstrated the feasibility of the methodology that the research addressed.

The presented CAD-ISE relies on the integration of CAD and simulation through the application of object-oriented concepts to provide an integrated environment that will allow practitioners to use simulation without extensive knowledge of simulation or large amounts of data entry. Although the proposed research focuses on specific construction operations, the methodology and strategy developed here can also be applied to other applications. Furthermore, the classes developed here can be used to implement these applications with minor modifications and the addition of more construction process simulation models to the library.

8.2 Research Contributions

The research addressed a new methodology to conduct simulation. The system developed in this research achieved the integration of CAD with simulation successfully. It provided a solution to simplify building construction simulation models, and the possibility of integrating with other construction applications, such as data acquisition, animation, scheduling, and estimating. This research has resulted in the following major contributions:

1. Development of the methodology to integrate a CAD system with simulation. The populator model developed in the research is the first research effort to create Product Atomic Components (PACs) from the CAD drawing for the purpose of simulation. The PAC acts as an intelligent repository of information that integrates the CAD system with simulation.
2. Development of the Product-oriented simulation methodology. The research addresses an alternative way to design, build, and conduct construction simulation. It provides solutions to most obstacles that traditional simulation methodology faced, such as a large amounts of information entry, the complexity of simulation models, integration with other applications, and intelligent information repository.
3. Development of a combination of Product-oriented simulation and Process-oriented simulation. The research successfully employs a Process-oriented simulation engine to conduct Product-oriented simulation. This led to the prototype system CAD-ISE, which takes full advantage of both Product-oriented simulation and Process-oriented simulation.
4. Development of a construction simulation template. The research simplifies construction simulation model building by providing a pre-defined common construction simulation template, and offers a

straightforward way to assembly these pre-defined modeling elements in template.

5. Demonstration of the feasibility of integrating other applications in construction. The intelligent repository of information facilitates the development of integrated construction applications. Data acquisition, animation, scheduling, and estimating can be naturally integrated with CAD-ISE, which already has estimating and scheduling features.

8.3 Recommendations

This thesis presented the framework and methodology for CAD-ISE that will eventually become an entire CAD-based Integration Simulation Environment. The whole system exceeds the scope of one Ph.D. effort, so this thesis focused solely on the CAD-populator, the Product-oriented simulation concept, and the CAD-based simulation environment prototype system. Recommendations for further study follow:

1. Implement the entire CAD-based Integrated Simulation Environment. The prototype system presented in this thesis accomplished most aspects of a CAD-ISE. However, adding more models, such as animation and data acquisition, will make CAD-ISE more useful.
2. Implement a more efficient simulation engine. The current simulation engine is sluggish, and a new efficient engine is needed to simulate larger construction projects.
3. Develop the environment using an object-oriented programming language. A major portion of the current version of CAD-ISE was developed in Visual Basic. As it is not an object-oriented programming language, CAD-

ISE cannot fully take advantage of the object-oriented methodology. Using an object-oriented programming language will make the system more flexible and powerful.

4. Apply the Product-oriented simulation methodology to other kinds of construction simulation projects. The current research provides a solution to heavy construction simulation using Product-oriented simulation methodology; extending this methodology to other kinds of construction simulation requires further study.
5. Model more real world construction situations, such as weather and resource broken down, which makes the simulation model more close to real construction process.

REFERENCES

AbouRizk, S., and Hajjar, D. (1998). "A Framework for Applying Simulation in the Construction Industry." *Canadian Journal of Civil Engineering*, 25(3), 604-617.

Alfares, M., and Seireg, A. (1996). "An integrated system for computer-aided design and construction of reinforced concrete building using modular forms." *Automation in Construction*, 5(4), 323-341.

Chang, D. Y. (1991). "Object-oriented simulation system for construction process planning." *Construction Congress 91*, ASCE, New York, NY, 626-631.

Chang, D. Y., and Carr, R. I. (1987). "RESQUE: A Resource Oriented Simulation System for Multiple Resource Constrained Processes." *Proceedings of the PMI Seminar/Symposium*, Milwaukee, Wisconsin, 4-19.

Cherneff, J., Logcher, R., and Sriram, D. (1991). "Integrating CAD with Construction Schedule Generation." *Journal of Computing in Civil Engineering*, ASCE, 5(1), 64-83.

Chin, S., Liu, L. Y., Stumpf, A. L., Ganeshan, R., and Hicks, D. K. (1995). "CADD-Based Construction information management for Corps of Engineers Projects." *Computing in Civil Engineering: Proceedings of the Second Congress*, ASCE, New York, NY, 187-194.

Dharwadkar, P. V., and Gatton, T. M. (1995). "An Object-Oriented Building Model for CAD/Schedule Integration." *Computer in Civil Engineering: Proceedings of the Second Congress*, ASCE, New York, NY, 1248-1251.

Fisher, M. A., and Aalami, F. (1996). "Scheduling with Computer-Interpretable Construction Method Models." *Journal of Construction Engineering and Management*, 122(4), 337-347.

Hajjar, D., and AbouRizk, S. (1998). "Modeling and Analysis of Aggregate Production Operations." *Journal of Construction Engineering and Management*, ASCE, 124(5), 390-401.

Hajjar, D., and AbouRizk, S. (1996). "Building a Special Purpose Simulation Tool for Earthmoving Operations." *Proceedings of the 1996 Winter Simulation Conference*, ASCE, 1313-1320.

Hajjar, D., AbouRizk, S., and Xu, J. (1998). "Optimizing Construction Site Dewatering Operations using CSD." *Canadian Journal of Civil Engineering*, 25(5), 819-828.

Huang, R., and Halpin, D. W. (1995). "Recent Advancements in Simulation User Interface – A Description of the DISCO Environment." *Computing in Civil Engineering: Proceedings of the Second Congress*, ASCE, New York, NY, 558-566.

Ioannou, P. G., and Martinez, J. (1996). "Animation of Complex Construction Simulation Models." *Computing in Civil Engineering: Proceedings of the Third Congress*, ASCE, New York, NY, 620-626.

Ito, K. (1998). "Utilization of 3-D Graphical Simulation with Object-Oriented Product Model for Building Construction Process." *Computing in Civil Engineering: Proceedings of the International Computing*, ASCE, Reston, VA, 73-83.

Kartam, N. A. (1994). "ISICAD: Interactive System for Integrating CAD and Computer-Based Construction Systems." *Microcomputers in Civil Engineering*, 9, 41-51.

Kim, I., Liebich, T., and Maver, T. (1997). "Managing design data in an integrated CAD environment: a product model approach." *Automation in Construction*, 7(1), 35-53.

McCahill, D. F., and Bernold, L. E. (1993). "Resource-Oriented Modeling and Simulation in Construction." *Journal of Construction Engineering and Management*, ASCE, 119(3), 590-605.

McKinney, K., and Fischer, M. (1997). "4D Analysis of Temporary Support." *Computing in Civil Engineering: Proceedings of the Fourth Congress*, ASCE, New York, NY, 470-476.

Nguyen, T. H., and Oloufa, A. A. (1995). "Construction Planning of Hi-Rise Buildings Object-Oriented Solid Geometry Model." *Computing in Civil Engineering: Proceedings of the Second Congress*, ASCE, New York, NY, 426-431.

Oloufa, A. A., and Ikeda, M. (1995). "Library-Based Simulation Modeling in Construction." *Computing in Civil Engineering: Proceedings of the Second Congress*, ASCE, New York, NY, 198-205.

Op den Bosch, A., and Baker, N. (1995). "Simulation of Construction Operations in Virtual Interactive environments." *Computing in Civil Engineering: Proceedings of the Second Congress*, ASCE, New York, NY, 1435-1442.

Sadonio, M., and Tommelein, I. D. (1998). "The Last Designer's CAD-Database for Sourcing, Procurement, and Planning." *Computing in Civil Engineering: Proceedings of the International Computing Congress*, ASCE, Reston, VA, 364-375.

Skolnick, J. F. (1993). "A CAD-Based Construction Simulation Tool Kit for Construction Planning." *Computing in Civil and Building Engineering: Proceedings of the Fifth International Conference*, ASCE, New York, NY, 117-124.

Teicholz, P., and Fischer, M. (1994). "Strategy for Computer Integrated Construction Technology." *Journal of Construction Engineering and Management*, 120(1), ASCE, 117-131.

Tommelein, I.D., Carr, R. I., and Odeh, A. M. (1994). "Assembly of Simulation Networks Using Design, Plans, and Methods." *Journal of Construction Engineering and Management*, ASCE, 120(4), 796-815.

Xu, J., and AbouRizk, S. (1999). "Product-based Model Representation for Integrating 3D CAD with Computer Simulation." *Winter Simulation Conference*, Phoenix, Arizona, 971-977.

Xu, J., and AbouRizk, S. (1999). "CAD-based Integrated Simulation Environment." *Third Canadian Construction Specialty Conference*, CSCE, Regina, Sask., 37-46.

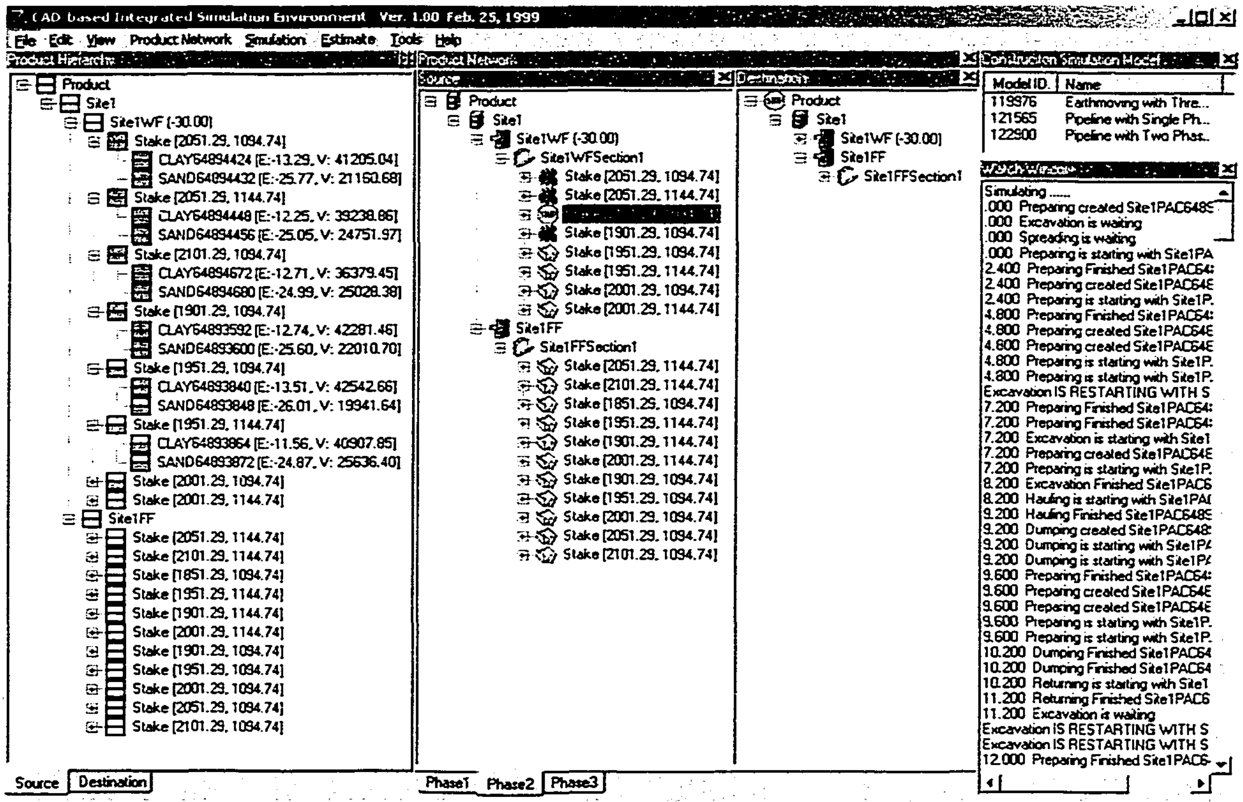
Williams, M. (1996). "Graphical Simulation for Project Planning: 4D-PlannerTM." *Computing in Civil Engineering: Proceedings of the Third Congress*, ASCE, Anaheim, CA, 404-409.

APPENDIX 1 CAD-ISE USER MANUAL

The main function of the CAD-ISE is the integration of CAD and Simulation, which provides an environment in which to conduct CAD-based simulation.

1. User interface

Fig. A1-1 illustrates the CAD-ISE user interface.


CAD based Integrated Simulation Environment Ver. 1.00 Feb. 25, 1999
File Edit View Product Network Simulation Estimate Tools Help
Product Hierarchy
Product Network
Source
Destination
Construction Simulation Model
Watch Window
Source Destination
Phase1 Phase2 Phase3

```
Product
  Site1
    Site1WF (-30.00)
      Stake [2051.29, 1094.74]
        CLAY64894424 [E:-13.29, V: 41205.04]
        SAND64894432 [E:-25.77, V: 21160.68]
      Stake [2051.29, 1144.74]
        CLAY64894448 [E:-12.25, V: 39238.86]
        SAND64894456 [E:-25.05, V: 24751.97]
      Stake [2101.29, 1094.74]
        CLAY64894672 [E:-12.71, V: 36379.45]
        SAND64894680 [E:-24.99, V: 25028.38]
      Stake [1901.29, 1094.74]
        CLAY64893592 [E:-12.74, V: 42281.46]
        SAND64893600 [E:-25.60, V: 22010.70]
      Stake [1951.29, 1094.74]
        CLAY64893840 [E:-13.51, V: 42542.66]
        SAND64893848 [E:-26.01, V: 19941.64]
      Stake [1951.29, 1144.74]
        CLAY64893864 [E:-11.56, V: 40907.85]
        SAND64893872 [E:-24.87, V: 25636.40]
      Stake [2001.29, 1094.74]
      Stake [2001.29, 1144.74]
    Site1FF
      Stake [2051.29, 1144.74]
      Stake [2101.29, 1144.74]
      Stake [1851.29, 1094.74]
      Stake [1951.29, 1144.74]
      Stake [1901.29, 1144.74]
      Stake [2001.29, 1144.74]
      Stake [1901.29, 1094.74]
      Stake [1951.29, 1094.74]
      Stake [2001.29, 1094.74]
      Stake [2051.29, 1094.74]
      Stake [2101.29, 1094.74]
```

```
Product
  Site1
    Site1WF (-30.00)
      Site1WFSection1
        Stake [2051.29, 1094.74]
        Stake [2051.29, 1144.74]

        Stake [1901.29, 1094.74]
        Stake [1951.29, 1094.74]
        Stake [1951.29, 1144.74]
        Stake [2001.29, 1094.74]
        Stake [2001.29, 1144.74]
    Site1FF
      Site1FFSection1
        Stake [2051.29, 1144.74]
        Stake [2101.29, 1144.74]
        Stake [1851.29, 1094.74]
        Stake [1951.29, 1144.74]
        Stake [1901.29, 1144.74]
        Stake [2001.29, 1144.74]
        Stake [1901.29, 1094.74]
        Stake [1951.29, 1094.74]
        Stake [2001.29, 1094.74]
        Stake [2051.29, 1094.74]
        Stake [2101.29, 1094.74]
```

```
Product
  Site1
    Site1WF (-30.00)
    Site1FF
      Site1FFSection1
```

Model ID.	Name
119976	Earthmoving with Thre...
121565	Pipeline with Single Ph...
122900	Pipeline with Two Phas...

```
Simulating ......
.000  Preparing created Site1PAC6489
.000  Excavation is waiting
.000  Spreading is waiting
.000  Preparing is starting with Site1PA
2.400  Preparing Finished Site1PAC64:
2.400  Preparing created Site1PAC64E
2.400  Preparing is starting with Site1P.
4.800  Preparing Finished Site1PAC64:
4.800  Preparing created Site1PAC64E
4.800  Preparing created Site1PAC64E
4.800  Preparing is starting with Site1P.
4.800  Preparing is starting with Site1P.
Excavation IS RESTARTING WITH S
7.200  Preparing Finished Site1PAC64:
7.200  Preparing Finished Site1PAC64:
7.200  Excavation is starting with Site1
7.200  Preparing created Site1PAC64E
7.200  Preparing is starting with Site1P.
8.200  Excavation Finished Site1PAC6
8.200  Hauling is starting with Site1PA(
9.200  Hauling Finished Site1PAC6489
9.200  Dumping created Site1PAC648:
9.200  Dumping is starting with Site1P/
9.200  Dumping is starting with Site1P/
9.600  Preparing Finished Site1PAC64:
9.600  Preparing created Site1PAC64E
9.600  Preparing created Site1PAC64E
9.600  Preparing is starting with Site1P.
9.600  Preparing is starting with Site1P.
10.200  Dumping Finished Site1PAC64
10.200  Dumping Finished Site1PAC64
10.200  Returning is starting with Site1
11.200  Returning Finished Site1PAC6
11.200  Excavation is waiting
Excavation IS RESTARTING WITH S
Excavation IS RESTARTING WITH S
12.000  Preparing Finished Site1PAC6-
```

Fig. A1 - 1 CAD-ISE User Interface

- The Product Hierarchy window shows both source PH and Destination PH. The icon of each item in the hierarchy displays the status of each phase; shaded indicates completion, and blank indicates that the phase is not yet done.

- The Product Network window includes all construction phases. For each phase, there is a Source window, which defines the source site construction sequence, and a Destination window, which defines the destination site construction sequence.
- The Construction Simulation Model window lists all available simulation models.
- The Watch window gives simulation-tracing information.

2. Main Menu

The menu bar at the top of the form is the command center of CAD-ISE. Each option on the menu bar activates a drop down menu of commands that the user can utilize to create, modify, and experiment with a simulation project.

- **File Menu**. The file menu includes commands that can be used to create a new project, open an existing project, and save a project.
- **Edit Menu**. Not available in this version.
- **View Menu**. The view menu includes commands that can be used to show or hide the Product Hierarchy window, Construction Phase window (i.e. Product Network Window), Simulation Model window, and Tracer window (i.e. Watch Window).
- **Product Network Menu**. The product network menu includes commands that can be used to add new construction phases, add multiple sites, and create new sections for both the source site and destination site.
- **Simulation Menu**. The simulation menu includes commands that can be used to initialize and run the simulation. The initialize command brings up a dialog box

that can be used to create the linkage between each construction phase and the simulation model.

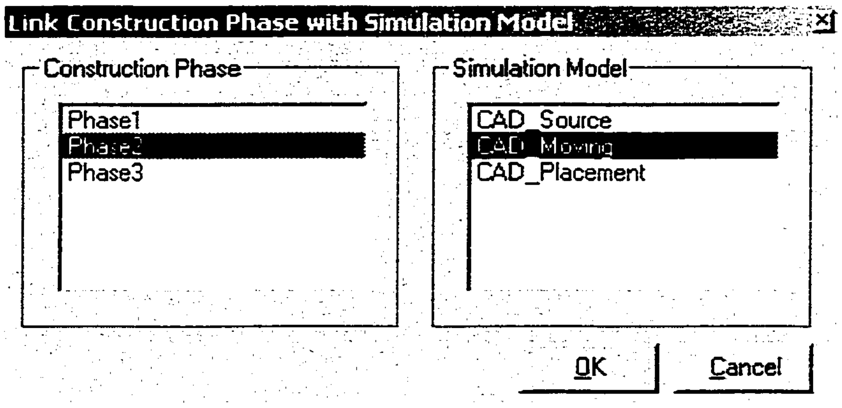


Fig. A1 - 2 Simulation Initialization Window

- **Estimate Menu.** The estimate menu is used to generate estimation reports based on the simulation results, including product estimate, activity estimate, and detail resource estimate.
- **Tools Menu.** The tools menu is the central menu used to communication with other applications and set options, including launching other applications such as AutoCAD, Simphony Designer, and Simphony Editor, checking information, such as the soil type in Product Model and resource availability, and setting options to control the display of detailed information.
- **Help Menu.** The help menu is unavailable in this version.

3. Dialog Boxes

The dialog box menus are used in the Product Network window to create, modify, and query the Product Network.

The dialog box items vary from level to level. At the form level, items include *Show Relationship*, *Show Predecessors, Show Successors*, and *Link CAD,* which activates the objects in AutoCAD. At the product level, items include *Show* and *Hide* PACs, *Add Site, AddAs Predecessor, AddAs Sucessor, Delete Relationship,* and *Delete.* At the site level, *Add Dummy Working Space* replaces the *Add Site* of the product level. At the working space level, *Add Dummy Section* replaces the *Add Dummy Working Space* of the previous level. At the section level, *Add Stakes* replaces the *Add Dummy Section* of the working space level. At the PAC level, *PAC Parameters* replaces the *Add Stakes* of the section level.

4. Quick Start

Chapter 5 provides a case study that details a quick start to build a simulation project under CAD-ISE.

APPENDIX 2 CAD-ISE MAJOR MODELING ELEMENT CODE

Modeling Element: ***<u>CAD Entity</u>***

```
Option Explicit

Public Function CAD_Entity_OnCreate(ob As CFCSim_ModelingElementInstance, x As Single, y As Single) As Boolean
  ob.OnCreate x,y,True
  CAD_Entity_OnCreate=True

  'CAD_ISE Begin
  ob.AddAttribute "BeforeRef", "Product Object Reference before action", CFC_Object, CFC_Single, CFC_Hidden
  ob.AddAttribute "AfterRef", "Destination Product Object Reference after action", CFC_Object, CFC_Single, CFC_Hidden
  'CAD_ISE End

  ob.AddAttribute "Quantity", "Number of Entities to Create", CFC_Numeric, CFC_Single, CFC_ReadWrite,1,1000
  ob.AddAttribute "First","Time of First Create",CFC_Numeric, CFC_Single, CFC_ReadWrite,0,1000000
  ob.AddAttribute "Between","Time Between Creates",CFC_Distribution, CFC_Single,CFC_ReadWrite
  ob.AddAttribute "Fired","Entites created so far",CFC_Numeric, CFC_Single,CFC_Hidden

  ob.AddAttribute "Image", "Image", CFC_Text, CFC_ListBox, CFC_ReadWrite
  ob.AddAttribute "Name", "Name", CFC_Text, CFC_ListBox, CFC_ReadWrite
  ob.AddAttribute "txtType", "Descriptive Type", CFC_Text, CFC_Single, CFC_ReadWrite
  ob.AddAttribute "Priority","Process Priority",CFC_Numeric, CFC_Single, CFC_ReadWrite,1
  ob.AddAttribute "Mode", "Product Mode", CFC_Text, CFC_ListBox, CFC_ReadWrite
  ob.AddAttribute "Capacity", "Capacity", CFC_Numeric, CFC_Single, CFC_ReadWrite,1,1000000
  ob.AddAttribute "Tolerance", "Tolerance", CFC_Numeric, CFC_Single, CFC_ReadWrite,0,1000000

  ob("Quantity")=1
  ob("First")=0
  ob("Between")=0
  ob("Fired") = 0

  ob("Image") = "EMS_Truck"
  ob("Name")="Caterpillar Model 777C"
  ob("txtType") = "None"
  ob("Priority") = 1
  ob("Mode") = "Volume"
  ob("Capacity") = 45
  ob("Tolerance") = 0

  ob.AddStatistic "CycleTime",ob("Name") & " Cycle Time",False,True

  ob.AddConnectionPoint "Out",x+58,y+25,COutput,5
End Function

Public Sub CAD_Entity_OnDragDraw(ob As CFCSim_ModelingElementInstance)
  ob.OnDraw
End Sub

Public Sub CAD_Entity_OnDraw(ob As CFCSim_ModelingElementInstance)
  If Len(ob("Image")) > 0 Then CDC.RenderPicture ob("Image"),ob.CoordinatesX(0)+8,ob.CoordinatesY(0)+8,33,33
  If ob.Selected Then CDC.ChangeLineStyle CFC_SOLID,3,RGB(0,0,255)

  CDC.Circ ob.CoordinatesX(0)+25,ob.CoordinatesY(0)+25,25
  CDC.TextOut ob.CoordinatesX(0)+10,ob.CoordinatesY(0)+20, ob("Quantity")

  ob.DrawConnectionPoints
End Sub

Public Sub CAD_Entity_OnListBoxInitialize(ob As CFCSim_ModelingElementInstance, attr As CFCSim_Attribute, lstList As
Object)
Dim myset As Recordset
  Select Case attr.Name
    Case "Image"
      Set myset = SimphonyConnection.OpenRecordset("Select * From STL_Bitmaps", dbOpenSnapshot)
```

```
            myset.MoveFirst
            While Not myset.EOF
              lstlist.AddItem myset!Designation
              myset.MoveNext
            Wend
           myset.Close
        Case "Name"
           Set myset = SimphonyConnection.OpenRecordset("Select * From EMS_TruckTypes", dbOpenSnapshot)
           myset.MoveFirst
           While Not myset.EOF
             lstlist.AddItem myset!Description
             myset.MoveNext
           Wend
           myset.Close
        Case "Mode"
           lstlist.AddItem "Piece"
           lstlist.AddItem "Volume"
           lstlist.AddItem "Area"
           lstlist.AddItem "Length"
    End Select
End Sub

Public Function CAD_Entity_OnRelationValid(srcCP As CFCSim_ConnectionPoint, dstCP As CFCSim_ConnectionPoint) As
Boolean
   CAD_Entity_OnRelationValid=True

   If srcCP.RelationsTo.Count>0 Then
      MessagePrompt "Only one relation is allowed from this connection point "
      CAD_Entity_OnRelationValid=False
   End If
End Function

Public Sub CAD_Entity_OnSimulationInitialize(ob As CFCSim_ModelingElementInstance)
   ob.AddEvent "FireEntity"
End Sub

Public Sub CAD_Entity_OnSimulationInitializeRun(ob As CFCSim_ModelingElementInstance, RunNum As Integer)
      ob.ScheduleEvent ob.AddEntity,"FireEntity",ob("First")
   ob("Fired")=0
End Sub

Public Sub CAD_Entity_OnSimulationProcessEvent(ob As CFCSim_ModelingElementInstance, MyEvent As String, Entity As
CFCSim_Entity)
   If ob("Fired")>= ob("Quantity") Then Exit Sub
   ob("Fired")=ob("Fired")+1

Dim newEntity As CFCSim_Entity
   Set newEntity =  ob.AddEntity

'CAD_ISE Begin
Dim SimPM As New CADISE_SimPM
   Set SimPM.Source = ob("BeforeRef").Reference
   Set SimPM.Destination =  ob("AfterRef").Reference
   Select Case ob("Mode")
      Case "Piece"
         SimPM.Mode = 0
      Case "Volume"
         SimPM.Mode = 1
      Case "Area"
         SimPM.Mode = 2
      Case "Length"
         SimPM.Mode = 3
   End Select
   SimPM.Capacity = ob("Capacity")
   SimPM.Tolerance = ob("Tolerance")
   Set newEntity("PM") = SimPM
'CAD_ISE End

   newEntity("obid")=ob.Id
```

```
    newEntity("Name")=ob("Name")
    newEntity("Capacity")=ob("Capacity")
    newEntity("Dummy") = 0
    newEntity("txtType")=ob("txtType")
    newEntity("Priority")=ob("Priority")

    Set newEntity("CycleStat")= ob.stat("CycleTime")
    newEntity("PayLoad")=0
    newEntity("StartTime") = -1

    'set specific parameters for truck
    If ob("Image") = "EMS_Truck" Then
      Dim myset As Recordset
      Set myset = SimphonyConnection.OpenRecordset("select * From EMS_TruckTypes where description ='" _
                              & ob("Name") & "'", dbOpenSnapshot)
      newEntity("MeanL")=myset!MeanL
      newEntity("StDevL")=myset!StDevL
      newEntity("ScaleFactorL")=myset!ScaleFactorL
      newEntity("YFactorL")=myset!YFactorL
      newEntity("MeanE")=myset!MeanE
      newEntity("StDevE")=myset!StDevE
      newEntity("ScaleFactorE")=myset!ScaleFactorE
      newEntity("YFactorE")=myset!YFactorE
      myset.Close
    End If

    ob.TransferOut newEntity
    Tracer.Trace ob("Name") & " created and transferred: " & newEntity.Id , "Simulation"
    ob.ScheduleEvent Entity, "FireEntity", ob("Between")
  End Sub
```

Modeling Element: CAD Activity

```
Option Explicit

Public Function CAD_Activity_OnCreate(ob As CFCSim_ModelingElementInstance, x As Single, y As Single) As Boolean
ob.OnCreate x,y,True
CAD_Activity_OnCreate=True

  ob.AddAttribute "Image", "Image", CFC_Text, CFC_ListBox, CFC_ReadWrite
  ob.AddAttribute "Name", "Name", CFC_Text, CFC_Single, CFC_ReadWrite
  ob.AddAttribute "ActivityID", "Activity ID", CFC_Numeric, CFC_Single, CFC_ReadWrite,1
  ob.AddAttribute "Governing", "Governing Activity", CFC_Text, CFC_ListBox, CFC_ReadWrite
  ob.AddAttribute "PM_Created", "PM Created from", CFC_Text, CFC_ListBox, CFC_ReadWrite
  ob.AddAttribute "Site", "Site Number", CFC_Numeric, CFC_Single, CFC_ReadWrite
  ob.AddAttribute "PH", "Update PH", CFC_Text, CFC_ListBox, CFC_ReadWrite
  ob.AddAttribute "Statistic", "Statistic Collection", CFC_Text, CFC_ListBox, CFC_ReadWrite
  ob.AddAttribute "SubAct", "Activate Sub-Activies", CFC_Text, CFC_ListBox, CFC_ReadWrite
  ob.AddAttribute "Quantity", "Activity Quantity", CFC_Numeric, CFC_Single, CFC_ReadWrite
  ob.AddAttribute "Productivity", "Activity Productivity", CFC_Distribution, CFC_Single, CFC_ReadWrite
  ob.AddAttribute "Duration", "Activity Duration", CFC_Distribution, CFC_Single, CFC_ReadWrite

ob("Image") = "EMS_Excavator"
ob("Name") = "CAD Activity"
ob("ActivityID") = ob.Id
ob("Governing") = "Yes"
ob("PM_Created") = "None"
ob("Site") = 1
ob("PH") = "Source"
ob("Statistic") = "No"
ob("SubAct") = "No"
ob("Quantity") = 0

  With ob("Productivity").Distribution
    .DistType=CFC_Constant
    .ParameterValue(0)=0
  End With
  With ob("Duration").Distribution
  .DistType=CFC_Constant
    .ParameterValue(0)=0
  End With

  ob.AddConnectionPoint "In" , x-10, y+25, CInput, 5
  ob.AddConnectionPoint "Out",x+130,y+25,COutput,5

  ob.AddFile "Queue",QUEUE
  ob.AddFile "ResFile",QUEUE

End Function

Public Sub CAD_Activity_OnDragDraw(ob As CFCSim_ModelingElementInstance)
  ob.OnDraw
End Sub

Public Sub CAD_Activity_OnDraw(ob As CFCSim_ModelingElementInstance)
  If Len(ob("Image")) > 0 Then CDC.RenderPicture ob("Image"),ob.CoordinatesX(0),ob.CoordinatesY(0),120,50
  If ob.Selected Then CDC.ChangeLineStyle CFC_SOLID,3,RGB(0,0,255)

  CDC.Rectangle ob.CoordinatesX(0),ob.CoordinatesY(0),ob.CoordinatesX(0)+120,ob.CoordinatesY(0)+50
  CDC.TextOut ob.CoordinatesX(0)+5,ob.CoordinatesY(0)+20, ob("Name")

  ob.DrawConnectionPoints
End Sub

Public Sub CAD_Activity_OnListBoxInitialize(ob As CFCSim_ModelingElementInstance, attr As CFCSim_Attribute, lstList As
Object)
Dim myset As Recordset
  Select Case attr.Name
    Case "Image"
      Set myset = SimphonyConnection.OpenRecordset("Select * From STL_Bitmaps", dbOpenSnapshot)
```

```
        myset.MoveFirst
        While Not myset.EOF
          lstlist.AddItem myset!Designation
          myset.MoveNext
        Wend
        myset.Close
      Case "PM_Created"
          lstlist.AddItem "None"
          lstlist.AddItem "Source"
          lstlist.AddItem "Destination"
      Case "PH"
          lstlist.AddItem "Source"
          lstlist.AddItem "Destination"
          lstlist.AddItem "Both"
      Case "Governing","Statistic","SubAct"
          lstlist.AddItem "Yes"
          lstlist.AddItem "No"
  End Select
End Sub

Public Sub CAD_Activity_OnSimulationInitialize(ob As CFCSim_ModelingElementInstance)
  ob.AddEvent "Create",True
  ob.AddEvent "Allocate",True
  ob.AddEvent "Start",True
  ob.AddEvent "Finish"
  ob.AddEvent "Release",True
End Sub

Public Sub CAD_Activity_OnSimulationProcessEvent(ob As CFCSim_ModelingElementInstance, MyEvent As String, Entity As
CFCSim_Entity)
Dim childob As CFCSim_ModelingElementInstance
Dim res As CFCSim_ModelingElementInstance
Dim newEntity As CFCSim_Entity
Dim DumbEntity As CFCSim_Entity
Dim DummyEntity As Boolean
Dim Allocated As Boolean
Dim Created As Boolean
Dim Duration As Double
Dim I As Integer
  Select Case MyEvent
    Case "Create"  'Create Product Model
      Entity("PM").EngineID=ob.Id 'ob("ActivityID")
      Entity("PM").EngineType = ob.ElementType
      Entity("PM").EngineName = ob("Name")
      Created = True
      Select Case ob("PM_Created")
        Case "Source"
          Entity("PM").Site = ob("Site")
          Entity("PM").SPAC.Remain = Entity("PM").Capacity
          Entity("PM").SPAC.Clear
          Do While Entity("PM").SPAC.Remain > 0 And Not Entity("PM").SPAC.LastAvailablePM
            Call Entity("PM").Source.CreateSimPM(Entity("PM"),ob("Site"))
            If Not Entity("PM").SPAC.Created Then Exit Do
          Loop
          If Entity("PM").SPAC.Remain <= 0 Or Entity("PM").SPAC.LastPM Then
            For I = 1 To Entity("PM").SPAC.Count
              Entity("PM").Source.TracerMsg Format(SimTime,"###.000") & "  " & ob("Name") & " created " _
                                                  & Entity("PM").SPAC(I).PAC
            Next I
          Else
            Created = False
          End If
        Case "Destination"
          Entity("PM").Site = ob("Site")
          If Entity("PM").SPAC.Quantity > 0 Then
            Entity("PM").DPAC.Remain = Entity("PM").SPAC.Quantity
          Else
            Entity("PM").DPAC.Remain = Entity("PM").Capacity
          End If
          Entity("PM").DPAC.Clear
          Do While Entity("PM").DPAC.Remain > 0 And Not Entity("PM").DPAC.LastAvailablePM
            Call Entity("PM").Destination.DumpSimPM(Entity("PM"),ob("Site"))
            If Not Entity("PM").DPAC.Created Then Exit Do
          Loop
          If Entity("PM").DPAC.Remain <= 0 Or Entity("PM").DPAC.LastPM Then
            For I = 1 To Entity("PM").DPAC.Count
              Entity("PM").Destination.TracerMsg Format(SimTime,"###.000") & "  " & ob("Name") _
                                                          & " created " & Entity("PM").DPAC(I).PAC
            Next I
          Else
            Created = False
          End If
      End Select
      If Created Then
        ob.ScheduleEvent Entity,"Allocate",0
      Else
        Entity("PM").Source.TracerMsg  Format(SimTime,"###.000") & "  " & ob("Name") & " is waiting"
        ob.File("Queue").Add Entity
        Tracer.Trace "Entity-" & Entity.Id & " is adding to file", "Simulation"
      End If

    Case "Allocate"   'Allocate Resources
      If Entity("PM") Is Nothing Then
        DummyEntity = True
        ob.DeleteEntity Entity
      Else
        DummyEntity = False
        ob.File("ResFile").Add Entity
      End If
      Set Entity = CAD_Activity_GetEntity(ob)

      If Not Entity Is Nothing Then
        'Check whether all request resources are available
        Allocated = True
        For Each childob In ob.ChildElements
          If childob.ElementType = "CAD_Resource"  Then
            If childob("Allocated") = 0 Then
              Set res=childob("ResOb").Reference
              If res("Current") < childob("Quantity") Then
                Allocated = False
              ElseIf childob("Hold") = "Yes" Then
                childob("Allocated") = 1
                res("Current") = res("Current") - childob("Quantity")
                If DummyEntity And childob.File("ResFile").Length>0 Then
                  Call childob.DeleteEntity(childob.File("ResFile").Pop)
                End If
              End If
            End If
          End If
        Next childob

        'Allocate the resources
          For Each childob In ob.ChildElements
          If childob.ElementType = "CAD_Resource" Then
            If childob("Allocated") = 0 Then
              If Not DummyEntity Then
                Set newEntity = ob.CloneEntity(Entity)
                newEntity("NumRes") = childob("Quantity")
                childob.File("ResFile").Add newEntity, childob("Priority")
              End If
              Set res=childob("ResOb").Reference
              If childob.File("ResFile").Length > 0 Then
                childob.File("ResFile").MoveFirst
                If res("Current") >= childob.File("ResFile").Entity("NumRes") Then
                  If Allocated Then
                    childob("Allocated") = 1
                    res("Current") = res("Current") - childob.File("ResFile").Entity("NumRes")
                    Call childob.DeleteEntity(childob.File("ResFile").Pop)
                  End If
                End If
          End If
        End If
      End If
    Next childob

    If Allocated Then
      ob.File("ResFile").Remove Entity
      For Each childob In ob.ChildElements
        If childob.ElementType = "CAD_Resource" Then childob("Allocated") = 0
      Next childob
      ob.ScheduleEvent Entity, "Start", 0
    End If
  End If

Case "Start"
  Entity("PM").EntityID = CStr(Entity.Id)
  Entity("PM").SimTime = SimTime
  Select Case ob("PH")
    Case "None"
    Case "Source"
      For I = 1 To Entity("PM").SPAC.Count
        If Len(Entity("PM").SPAC(I).PAC) > 0 Then
          Entity("PM").Source.StartSimPM(Entity("PM"),Entity("PM").SPAC(I).PAC)
          Entity("PM").Source.TracerMsg Format(SimTime,"###.000") & "  " & ob("Name") _
                          & " is starting with " & Entity("PM").SPAC(I).PAC
        End If
      Next I
    Case "Destination"
      For I = 1 To Entity("PM").DPAC.Count
        If Len(Entity("PM").DPAC(I).PAC) > 0 Then
          Entity("PM").Destination.StartSimPM(Entity("PM"),Entity("PM").DPAC(I).PAC)
          Entity("PM").Destination.TracerMsg Format(SimTime,"###.000") & "  " & ob("Name") _
                          & " is starting with " & Entity("PM").DPAC(I).PAC
        End If
      Next I
    Case "Both"
      For I = 1 To Entity("PM").SPAC.Count
        If Len(Entity("PM").SPAC(I).PAC) > 0 Then
          Entity("PM").Source.StartSimPM(Entity("PM"),Entity("PM").SPAC(I).PAC)
          Entity("PM").Source.TracerMsg Format(SimTime,"###.000") & "  " & ob("Name") _
                          & " is starting with " & Entity("PM").SPAC(I).PAC
        End If
      Next I
      For I = 1 To Entity("PM").DPAC.Count
        If Len(Entity("PM").DPAC(I).PAC) > 0 Then
          Entity("PM").Destination.StartSimPM(Entity("PM"),Entity("PM").DPAC(I).PAC)
          Entity("PM").Destination.TracerMsg Format(SimTime,"###.000") & "  " & ob("Name") _
                          & " is starting with " & Entity("PM").DPAC(I).PAC
        End If
      Next I
  End Select
  If ob("Duration") > 0 Then
    Duration = ob("Duration")
  ElseIf ob("Productivity") > 0 Then
    If ob("Quantity") > 0 Then
      Duration = ob("Quantity")/ob("Productivity")
    Else
      Duration = Entity("PM").Quantity/ob("Productivity")
    End If
  Else
    Duration = 0
  End If
  For Each childob In ob.ChildElements
    If childob.ElementType = "CAD_Resource" Then
      childob("ResOb").Reference.ScheduleEvent ob.AddEntity,"CollectStat",0
    End If
  Next childob
  ob.ScheduleEvent Entity,"Finish",Duration

Case "Finish"
```

```
            Set Entity("PM").MEI = ob
            Set Entity("PM").Entity =Elements(CStr( Entity("obid")))
            Entity("PM").EntityID = CStr(Entity.Id)
            Entity("PM").SimTime = SimTime
            Select Case ob("PH")
              Case "None"
              Case "Source"
                For I = 1 To Entity("PM").SPAC.Count
                  If Len(Entity("PM").SPAC(I).PAC) > 0 Then
                    Entity("PM").Source.FinishSimPM(Entity("PM"),Entity("PM").SPAC(I).PAC)
                    Entity("PM").Source.TracerMsg Format(SimTime,"###.000") & "  " & ob("Name") _
                                    & " Finished " & Entity("PM").SPAC(I).PAC
                  End If
                Next I
              Case "Destination"
                For I = 1 To Entity("PM").DPAC.Count
                  If Len(Entity("PM").DPAC(I).PAC) > 0 Then
                    Entity("PM").Destination.FinishSimPM(Entity("PM"),Entity("PM").DPAC(I).PAC)
                    Entity("PM").Destination.TracerMsg Format(SimTime,"###.000") & "  " & ob("Name") _
                                    & " Finished " & Entity("PM").DPAC(I).PAC
                  End If
                Next I
              Case "Both"
                For I = 1 To Entity("PM").SPAC.Count
                  If Len(Entity("PM").SPAC(I).PAC) > 0 Then
                    Entity("PM").Source.FinishSimPM(Entity("PM"),Entity("PM").SPAC(I).PAC)
                    Entity("PM").Source.TracerMsg Format(SimTime,"###.000") & "  " & ob("Name") _
                                    & " Finished " & Entity("PM").SPAC(I).PAC
                  End If
                Next I
                For I = 1 To Entity("PM").DPAC.Count
                  If Len(Entity("PM").DPAC(I).PAC) > 0 Then
                    Entity("PM").Destination.FinishSimPM(Entity("PM"),Entity("PM").DPAC(I).PAC)
                    Entity("PM").Destination.TracerMsg Format(SimTime,"###.000") & "  " & ob("Name") _
                                    & " Finished " & Entity("PM").DPAC(I).PAC
                  End If
                Next I
            End Select
            Tracer.Trace ob("Name") & " completed", "Simulation"
            ob.ScheduleEvent Entity,"Release",0

          Case "Release"
            For Each childob In ob.ChildElements
              If childob.ElementType = "CAD_Resource" And childob("ResType") <> "Material" Then
                Set res=childob("ResOb").Reference
                res("Current")=res("Current") + Int(childob("Quantity"))
                For I = 1 To Int(childob("Quantity"))
                  Set DumbEntity = ob.AddEntity
                  Set DumbEntity("PM") = Nothing
                  res.ScheduleEvent DumbEntity, "Release",0
                Next I
              End If
            Next childob
            ob.TransferOut Entity,ob.ConnectionPoints("Out")
      End Select
    End Sub

    Public Sub CAD_Activity_OnSimulationTransferIn(ob As CFCSim_ModelingElementInstance, Entity As CFCSim_Entity, SrcCp As
    CFCSim_ConnectionPoint, DstCp As CFCSim_ConnectionPoint)
      If ob("SubAct") = "Yes" Then
        ob.OnSimulationTransferIn(Entity, SrcCp,DstCp,True)
      Else
        If ob("Statistic") = "Yes" Then
          If Entity("StartTime") > -1 Then Entity("CycleStat").Collect SimTime-Entity("StartTime")
          Entity("StartTime") = SimTime
        End If
        ob.ScheduleEvent Entity,"Create",0
      End If
    End Sub
```

```
Private Function CAD_Activity_GetEntity(ob As CFCSim_ModelingElementInstance) As CFCSim_Entity
Dim newEntity As CFCSim_Entity
  With ob.File("ResFile")
    If .Length > 0 Then
      .MoveFirst
      Set newEntity = .entity
      While Not .EOF
        If .entity("Priority") > newEntity("Priority") Then Set newEntity = .entity
        .MoveNext
      Wend
    End If
  End With
  Set CAD_Activity_GetEntity = newEntity
End Function
```

APPENDIX 3 POPULATOR CODE

Extractor.Def

```
//-------------------------------------------------------------//
//          Construction Engineering and Management            //
//               Civil Engineering Department                  //
//          University of Alberta, Canada, T6G 2G7             //
//                  1999/04/24 by Jianfei XU                   //
//-------------------------------------------------------------//

LIBRARY       Extractor
DESCRIPTION   'CAD_based Simulation Model -- Extractor'

EXPORTS
        acrxEntryPoint
        _SetacrxPtp
       acrxGetApiVersion
```

hCADSIM.h

```
//-------------------------------------------------------------//
//          Construction Engineering and Management            //
//               Civil Engineering Department                  //
//          University of Alberta, Canada, T6G 2G7             //
//                  1999/04/24 by Jianfei XU                   //
//-------------------------------------------------------------//

#ifndef _hCADSIM_h
#define _hCADSIM_h 1

#include <dbsol3d.h>
#include <adslib.h>

#define SERVICE  "simEntity_Service"


//========== simEntity Class ==========

class simEntity: public AcRxObject

{
public:

        ACRX_DECLARE_MEMBERS(simEntity);
        simEntity();
        virtual ~simEntity();

        virtual void GetFace3DPoints(ads_name FaceName);

protected:

private:

};

MAKE_ACDBOPENOBJECT_FUNCTION(simEntity);

#endif

//========== simEntity Class ==========
```

clsCADSIM.cpp

```
//-------------------------------------------------------------//
//             Construction Engineering and Management         //
//               Civil Engineering Department                  //
//          University of Alberta, Canada, T6G 2G7             //
//                  1999/04/24 by Jianfei XU                   //
//-------------------------------------------------------------//

//simEntity Class defines the class that will be used
```

```
//in the coming simulation model.

#include <rxregsvc.h>
#include <aced.h>
#include <dbents.h>
#include <dbsymtb.h>
#include <rxboiler.h>
#include <dbproxy.h>
#include <gemat3d.h>
#include <dbmain.h>
#include "hCADSIM.h"

// simEntity implementation:

ACRX_CONS_DEFINE_MEMBERS(simEntity, AcRxObject, 0);

simEntity::simEntity()
{
    rxInit();
}

simEntity::~simEntity()
{
    deleteAcRxClass(desc());
}

void simEntity::GetFace3DPoints(ads_name FaceName)
{
        struct resbuf *ebuf, *eb, *pRb, *pTemp;
        AcDbObject *FaceObject;
        AcDbObjectId eID;

        acdbGetObjectId(eID, FaceName);
        acdbOpenObject(FaceObject,eID,AcDb::kForRead);

        if (FaceObject->isKindOf(AcDbFace::desc()))
        {
                ebuf = ads_entget(FaceName);
                eb = ebuf;

                pRb = FaceObject->xData();

                if (pRb != NULL)
                {
                for (pTemp = pRb; pTemp->rbnext != NULL; pTemp = pTemp->rbnext)
                        {
                                ;
                        }
                } else
                {
                ads_regapp("SimData");
                pRb = ads_newrb(AcDb::kDxfRegAppName);
                pTemp = pRb;
                pTemp->resval.rstring = (char*) malloc(strlen("SimData") +1);
                strcpy(pTemp->resval.rstring, "SimData");
                }

                for (eb = ebuf; eb != NULL; eb = eb->rbnext)
                {
                if (eb->restype == 10 || eb->restype == 11 || eb->restype == 12 ||
                eb->restype == 13)
                        {
                                pTemp->rbnext = ads_newrb(1000 + eb->restype);
                                pTemp = pTemp->rbnext;
                                pTemp->resval.rpoint[X] = eb->resval.rpoint[X];
                                pTemp->resval.rpoint[Y] = eb->resval.rpoint[Y];
                                pTemp->resval.rpoint[Z] = eb->resval.rpoint[Z];
                        }
                }
                FaceObject->upgradeOpen();
                FaceObject->setXData(pRb);
                ads_relrb(ebuf);
                ads_relrb(pRb);
        }

        FaceObject->close();
        ads_printf(".");
}
```

Extractor.cpp

```
//------------------------------------------------------------//
//              Construction Engineering and Management        //
//                 Civil Engineering Department                //
//           University of Alberta, Canada, T6G 2G7            //
//                  1999/04/24 by Jianfei XU                   //
//------------------------------------------------------------//

#include <stdlib.h>
#include <string.h>
#include <rxregsvc.h>
#include <aced.h>
#include <dbsymtb.h>
#include <dbgroup.h>
#include "hCADSIM.h"

void initApp();
void unloadApp();
extern "C"      AcRx::AppRetCode acrxEntryPoint(AcRx::AppMsgCode, void*);

void Extractor();

void initApp()
{
    acrxBuildClassHierarchy();
    acrxRegisterService(SERVICE);

    acedRegCmds->addCommand("CAD_SIM",
        "CAD_SIM_Extractor", "Extractor", ACRX_CMD_MODAL,
        Extractor);
}

void unloadApp()
{
    acedRegCmds->removeGroup("CAD_SIM");

    delete acrxServiceDictionary->remove(SERVICE);
}

extern "C" AcRx::AppRetCode acrxEntryPoint(AcRx::AppMsgCode msg, void* pkt)
{
        switch (msg)
        {
                case AcRx::kInitAppMsg:
                        acrxDynamicLinker->unlockApplication(pkt);
                        initApp();
                        break;
                case AcRx::kUnloadAppMsg:
                        unloadApp();
    }
    return AcRx::kRetOK;
}

void Extractor()
{
        ads_name sset;

        if (ads_ssget(NULL, NULL, NULL, NULL, sset) != RTNORM) {return;}

        long ssLength;
        ads_sslength(sset, &ssLength);

        ads_printf("Preparing face models ......");

        for (long i = 0; i < ssLength; i++)
        {
                ads_name entName;
                ads_ssname(sset, i, entName);

                simEntity *simE = new simEntity();
                simE->GetFace3DPoints(entName);
                delete simE;
        }

        ads_printf("\nFace models are ready\n");

        ads_ssfree(sset);

}
```

APPENDIX 4 CAD-ISE CLASSES

<u>**3D Face Class Hierarchy in CAD Add-On Model**</u>

```
Face3Ds
 └ Face3D
    ├ObjectID
    ├Description
    ├cadLayer
    ├Face3DType
    │
    └Face3D (TIN3Ds)
       └ Tin3D
          ├ObjectID
          │
          ├Point(1) (Point)
          ├Point(2) (Point)
          └Point(3) (Point)
             ├X
             ├Y
             └Z
```

<u>**Grid Class Hierarchy in CAD Add-On Model**</u>

```
cadGrids
 └ cadGrid
    ├ObjectID
    ├X
    ├Y
    │
    ├WorkingLayers (Stakes)
    ├NaturalLayers (Stakes)
    └StakePoints (Stakes)
       └Stake
          ├ObjectID
          ├Description
          ├Tag
          └Elevation
```

Classes In SimPM (ActiveX)

```
CADISE_SimPM
 ├─EngineID
 ├─EngineName
 ├─EngineType
 ├─EntityID
 │
 ├─Site
 ├─Section
 ├─WorkingSpace
 ├─NaturalSurface
 │
 ├─SimTime
 ├─Dummy
 ├─Type
 ├─Mode
 ├─Capacity
 ├─Tolerance
 │
 ├─Source      ───► BridgeX.Reference   ───► SimSR in SimCAD
 ├─Destination ───► BridgeX.Reference   ───► SimDR in SimCAD
 │
 ├─MEI         ───► Modeling Element Instance in Simphony
 ├─Entity      ───► Entity in Simphony
 │
 ├─sPAC (CADISE_SimPACs)
 └─dPAC (CADISE_SimPACs)
     ├─Remain
     ├─LastPM
     ├─LastAvailablePM
     │
     └─CADISE_SimPAC
         ├─Quantity
         ├─Created
         ├─Done
         ├─StartEntity
         ├─FinishEntity
         │
         └─PAC (NetNode in SimCAD)
```

Classes In BridgeX (ActiveX)

```
BridgeX
├─TracerMsg
│
└─Reference
    ├─CreateSimPM
    ├─DumpSimPM
    ├─StartSimPM
    └─FinishSimPM
```

Classes In SimCAD (Main)

```
CADISE_SimulationModel
└─SimProject ───► Simphony.CFCSim_Project

CADISE_Simphases
├─sPH (CADISE_SimPH)
├─dPH (CADISE_SimPH)
│  ├─SimPMType
│  │
│  ├─Product (NetTree)
│  └─NetNodes
└─ CADISE_SimPhase
   ├─SimPID (Phase ID)
   ├─SimMID (Simulation Model ID)
   ├─SimMName (Simulation Model Name)
   │
   ├─SourceActivities (CADISE_Activity)
   ├─DestinationActivities (CADISE_Activity)
   │
   ├─SimSR (BridgeX -- ActiveX)
   ├─SimDR (BridgeX -- ActiveX)
   │
   ├─SimSP (CADISE_SimPN)
   └─SimDP (CADISE_SimPN)
      ├─SimPMType
      ├─SimPID
      ├─TracingEnabled
      ├─TracerBox (Common Control)
      │
      ├─Product (NetTree)
      └─NetNodes
```

NetTree

- Caption
- CaptionOn
- CaptionHeight
- Style
- Appearance
- BorderStyle
- Enabled
- Font
- HideSelection
- Indentation
- LabelEdit
- LineStyle
- MousePointer
- PathSeperator
- Sorted
- MouseIcon
- hW nd
- ImageList
- DropHighlight
- Parent (***Node***)
- AsChild
- AsSibling
- CopyButton
- MoveButton
- PopupMenus
- MenuItems
- PopupHitMenus
- Network
- PrevDropHighlight (***Node***)
- PopOption
- ***ACAD***
- SimP(***NetTree***)
- ***NetTree (TreeView)***
- ***Nodes***
- ***NetNodes***
- EVENTS
  - AfterLabelEdit
  - BeforeLabelEdit
  - Click
  - Collapse
  - DblClick
  - DragDropNode
  - DragOverNode
  - Expand
  - InitPopupHit
  - KeyDown
  - KeyPress
  - KeyUp
  - PopupHit
  - MouseDown
  - MouseMove
  - MouseUp
  - NodeClick

```
NetNodes
 └─NetNode
     ├─SimPID
     ├─SimPMType
     ├─CanStart
     ├─IsStarting
     ├─HasDone
     │
     ├─StartTime
     ├─FinishTime
     ├─Quantity
     ├─TotalCost
     │
     ├─Tag
     ├─Data
     ├─ImageIcon
     ├─SelectedImageIcon
     │
     ├─NetTree
     │
     ├─cad (acadPAC)
     │
     ├─Predecessors (NetNodes)
     ├─Successors (NetNodes)
     └─Statuses (CADISE_Statuses)

acadPAC
 ├─ObjectID
 ├─CursorID
 ├─Type
 ├─Volume
 ├─Quantity
 ├─Remain
 ├─cx
 ├─cy
 ├─cz
 ├─Length
 ├─Width
 └─Height

CADISE_Statuses
 └─CADISE_Status
     ├─PhaseID
     ├─PhaseName
     ├─Tag
     ├─CanStart
     ├─IsStarting
     ├─HasDone
     │
     └─Activities
         └─CADISE_Activity

CADISE_Activity
 ├─EngineID
 ├─ActivityID
 ├─ActOrder
 ├─EngineName
 │
 ├─Parameter (CADISE_ActPara)
 │   ├─FirstST
 │   ├─StartTime
 │   ├─TotalAT
 │   ├─FinishTime
 │   ├─LastFT
 │   ├─Governing
 │   ├─CanStart
 │   ├─IsStarting
 │   ├─HasDone
 │   └─ST
 │
 └─Resources
     └─CADISE_Resource
```

```
Crews
└─Crew
   ├─ID
   ├─Description
   │
   ├─Labs (Labours)
   │   └─Labour
   │      ├─ID
   │      ├─Description
   │      ├─LabourRate
   │      └─LabourOTRate
   │
   ├─Equs (Equipments)
   │   └─Equipment
   │      ├─ID
   │      ├─Description
   │      ├─Unit
   │      ├─LabourRate
   │      ├─FOGRate
   │      ├─RepairRate
   │      └─EquipmentRate
   │
   └─Rens (Rentals)
       └─Rental
          ├─ID
          ├─Description
          ├─Unit
          ├─LabourRate
          ├─FOGRate
          ├─RepairRate
          └─RentalRate

Materials
└─Material
   ├─ID
   ├─Description
   ├─Unit
   ├─UnitRate
   └─UnitFreightRate

Others
└─Other
   ├─ID
   ├─Description
   ├─Unit
   ├─LabourRate
   ├─FOGRate
   ├─RepairRate
   └─OtherRate
```

APPENDIX 5 LOADING POPULATER ADD-ON AUTOLISP CODE

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;; If you do not have an ACAD.LSP file on your system,
;; you must copy this file to AutoCAD directory.
;; Otherwise, add the following statements to the end of that file.
;; AutoCAD autoloading populater model
;;        1. loading extrator.arx
;; 2. loading CADISE.DBV
;;   Remark:
;; Make sure the above files are sitting in the correct directory
;; i.e. C:\CADISE\CADAddon
;;                Jianfei Xu, May 21, 1999
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

(defun C:loadapp()
   (arxload "C:\\CADISE\\CADAddOn\\Extractor.ARX")
   (setvar "FILEDIA" 0)
   (setvar "cmdecho" 0)
   (command "_VBALoad" "C:\\CADISE\\CADAddOn\\CADISE.DVB")
   (setvar "cmdecho" 1)
   (setvar "FILEDIA" 1)
   (princ)
)

(defun C:ChangeLayer()
   (setvar "FILEDIA" 0)
   (command "_-VBARun" "ChangeLayer")
   (setvar "FILEDIA" 1)
   (princ)
)

(defun C:CreateWorkingFaces()
   (setvar "FILEDIA" 0)
   (command "_-VBARun" "CreateWorkingFaces")
   (setvar "FILEDIA" 1)
   (princ)
)

(defun C:CreateGrids()
   (setvar "FILEDIA" 0)
   (command "_-VBARun" "CreateGrids")
   (setvar "FILEDIA" 1)
   (princ)
)

(defun C:CreateFinalFace()
   (setvar "FILEDIA" 0)
   (command "_-VBARun" "CreateFinalFace")
   (setvar "FILEDIA" 1)
   (princ)
)

(defun C:CreateNaturalFaces()
   (setvar "FILEDIA" 0)
   (command "_-VBARun" "CreateNaturalFaces")
   (setvar "FILEDIA" 1)
   (princ)
)

(defun C:LinearPM()
   (setvar "FILEDIA" 0)
   (command "_-VBARun" "LinearPM")
   (setvar "FILEDIA" 1)
   (princ)
)
```

APPENDIX 6 POPULATER AUTOCAD MENU SETUP

Insert the following lines into acad.mnu and compile it.

```
***POP8
**Populater
ID_SimCAD     [&Simulation Tools]
ID_SimLoad    [&Load]'_loadapp
              [--]
ID_SimCL      [&Change Layer]'_ChangeLayer
              [--]
ID_SimExtra   [&Populator]_extractor
              [--]
ID_SimNS      [Generate &Natural Surfaces]'_CreateNaturalFaces
ID_SimFS      [Generate &Final Surface]'_CreateFinalFace
ID_SimWS      [Create &Working Surfaces]'_CreateWorkingFaces
              [--]
ID_SimGrids   [Create &Grids]'_CreateGrids
              [--]
ID_LinearPM   [Create &Linear PACs]'_LinearPM
```