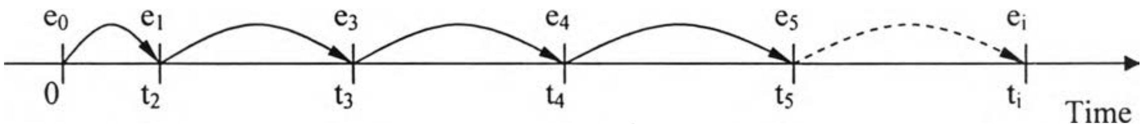
Chapter 5
System Development

5.1 Introduction

This chapter deals with the methodology of the Integrated System development. The methodology is divided into four parts. First, the visualization techniques of construction activities, which involve construction events and time are explained. Second, a simulation technique called "Monte Carlo simulation technique " is applied as a tool to develop the system. This simulation technique is presented in the second part. Next, methods of developing the Integrated System database are described. Finally, the tools and methods of Integrated System programming are illustrated.

5.2 Visualization of Construction Activities

In the Integrated System (IS), one important part of the system is the virtual scene. In this research, the virtual scene refers to the virtual system used for animating the virtual events of the construction activities in construction sites. The events of construction activities are the events used to illustrate the construction methods and techniques, for example, the methods and sequences of roof-structure installation. Those events present the different positions and paths of the 3D models of the construction machines, the positions of building structure and temporary work at the various construction times. The basic methods for creating the virtual system used for visualizing the construction activities are illustrated as follows.



e_i = The occurrence of the i^{th} event

t_i = The i^{th} point of time

Figure 5.1: The occurrences of the event at the point of time

5.2.1 The Events of Construction Activities

In the real world, there must be at least one event that occurs at a point of time. The occurrence of events may be stable or change when the time changes. Similar to the simulation modeling, the events will occur on the points of the simulation time as shown in Figure 5.1. The event e_1 occurs at the time t_1, when the time changes to t_2, the event e_2 occurs and the event e_i will occur, when the time changes to t_i, respectively.

The events of construction activities take place similar to the events in the real world. Pile-driving activity is one example of a construction activity that can be presented the occurrence of the events of construction activities.

At time t = t_0, the tower of pile-driving hammer works at X-position = x_0. When time changes to t = t_1 and t_2, it is moved along the X-axis direction to work at X-position = x_1 and x_2, respectively, as shown in the Figure 5.2.

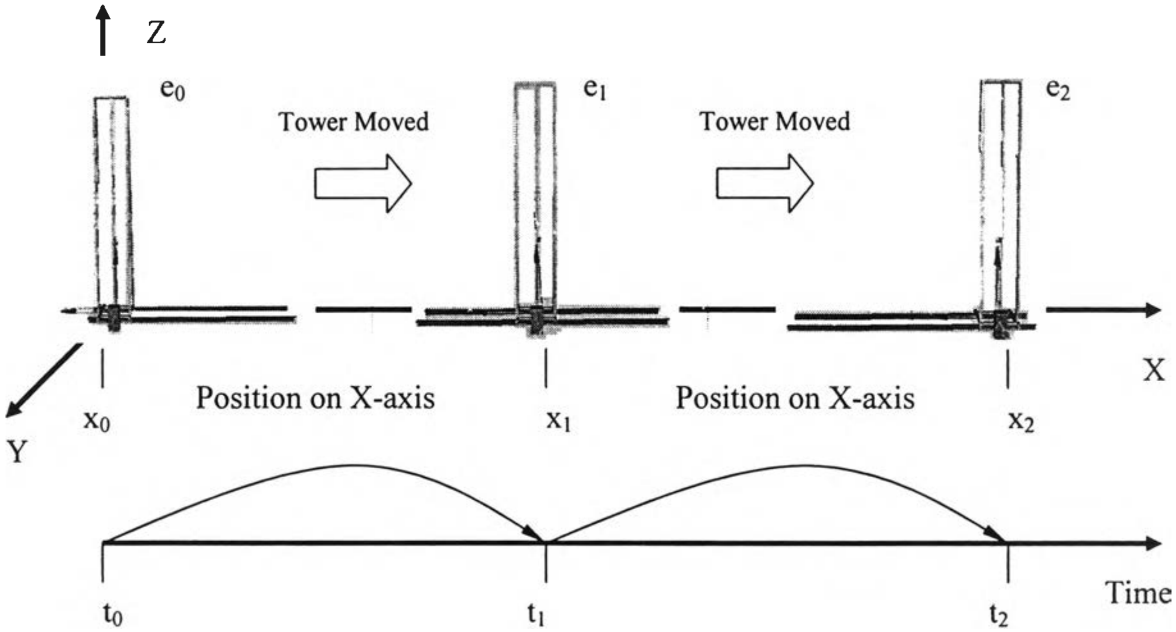


Figure 5.2: Events of pile-driving machine occur at different time advance.

5.2.2 The Controlling Parameters

On the virtual scene of the Integrated System, all objects in the construction-site must be controlled by controlling parameters. The controlling parameters are the parameters that are used as object movement controllers and used as the visualizer of the objects on the virtual scene. The most important controlling parameters for controlling the construction activities in this research consist of 1) transformation parameters, 2) linking parameters and 3) visibility parameters. The details of these controlling parameters are presented in Figure 5.3 and explained as follows.

5.2.2.1 Transformation Parameters

The transformation parameter is the parameter for controlling motion or movement of the objects in the virtual scene. The motion types of the objects can be divided in to 3 categories that are described by the following parameters:

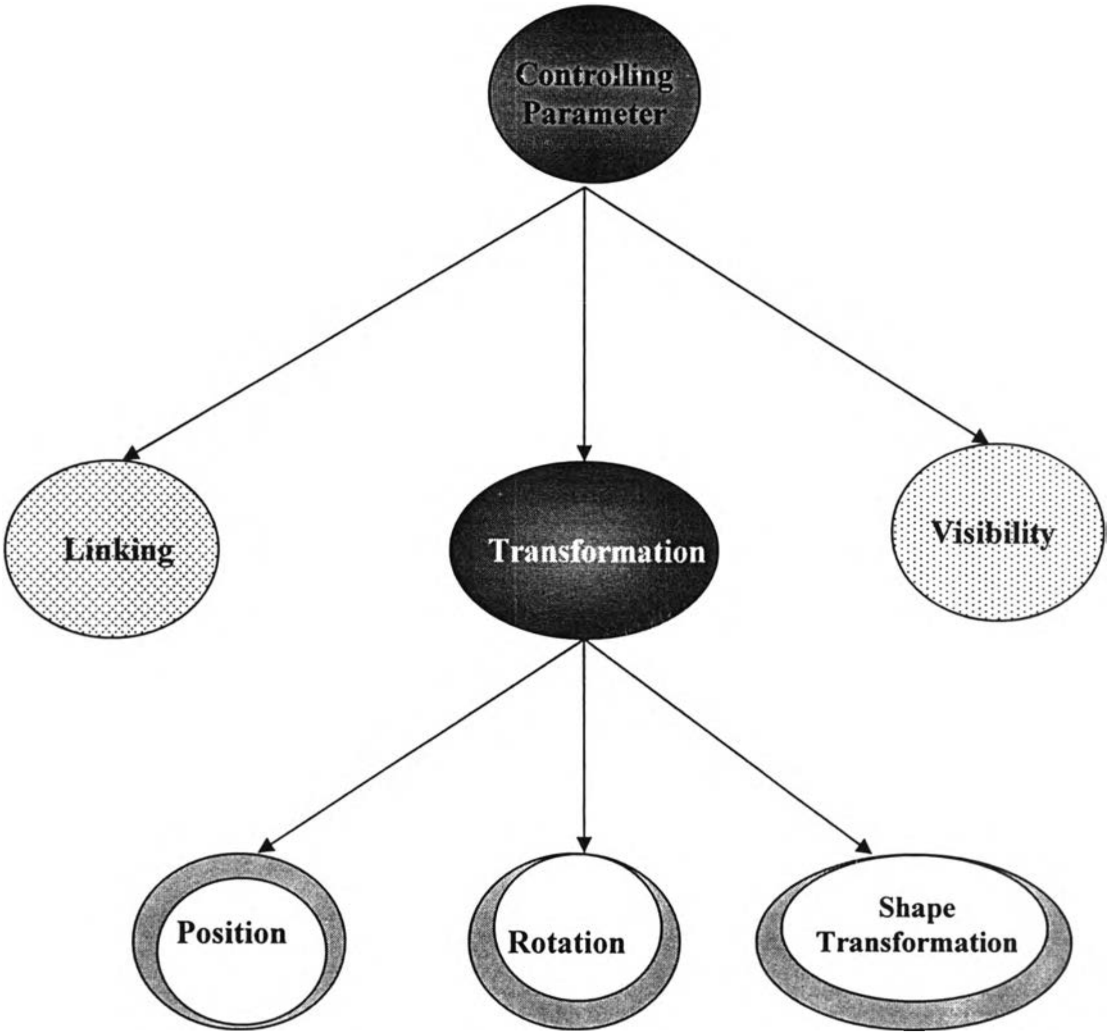


Figure 5.3: The controlling parameters

1) <u>Position Parameter</u>

The position parameter is the parameter used for controlling the object's movement in 3D space, such as movement along the X-axis, Y-axis and Z-axis direction, as shown in Figure 5.4-5.6. In Figure 5.4, the excavator's position was changed along the X-axis direction from x_0 to x_1 and from x_1 to x_2 while the time changed from t_0 to t_1 and from t_1 to t_2, respectively. In Figure 5.5, the truck-trailer was changed its position along the Y-axis direction from y_0 to y_1 and from y_1 to y_2 while the time changes from t_0 to t_1 and from t_1 to t_2, respectively. In Figure 5.6, the pile-driving hammer changed its position along the Z-axis direction from z_0 to z_1 and from z_1 to z_2 while the time changed from t_0 to t_1 and from t_1 to t_2, respectively.

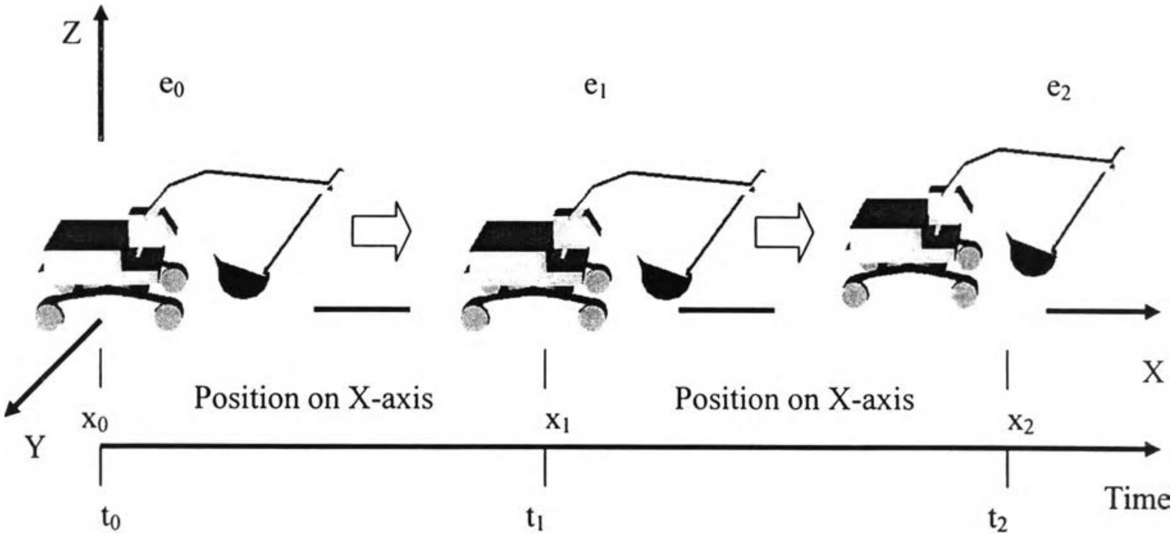


Figure 5.4: Excavator movement along the X-axis controlled by a position parameter

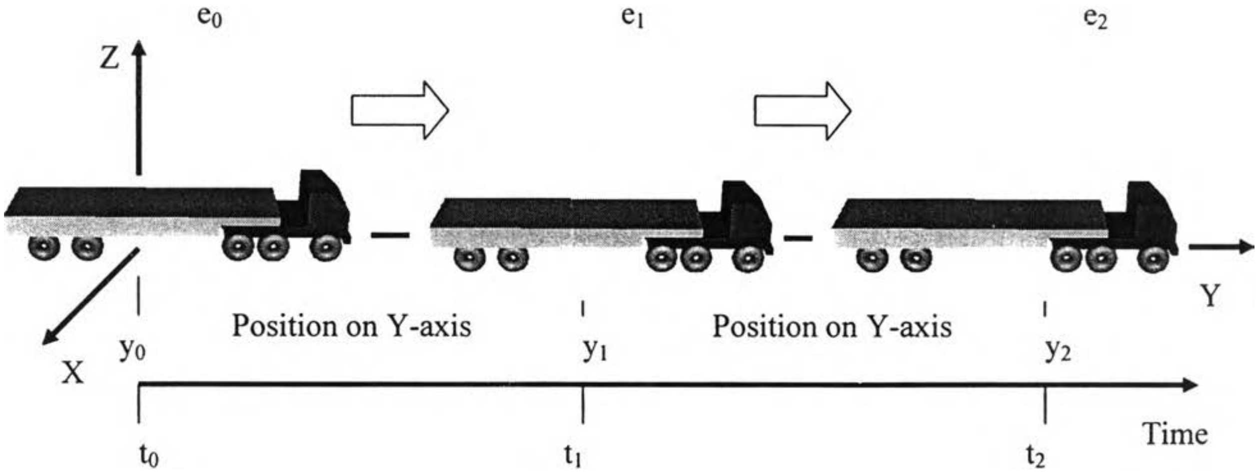


Figure 5.5: Tuck-trailer movement along the Y-axis controlled by a position parameter.

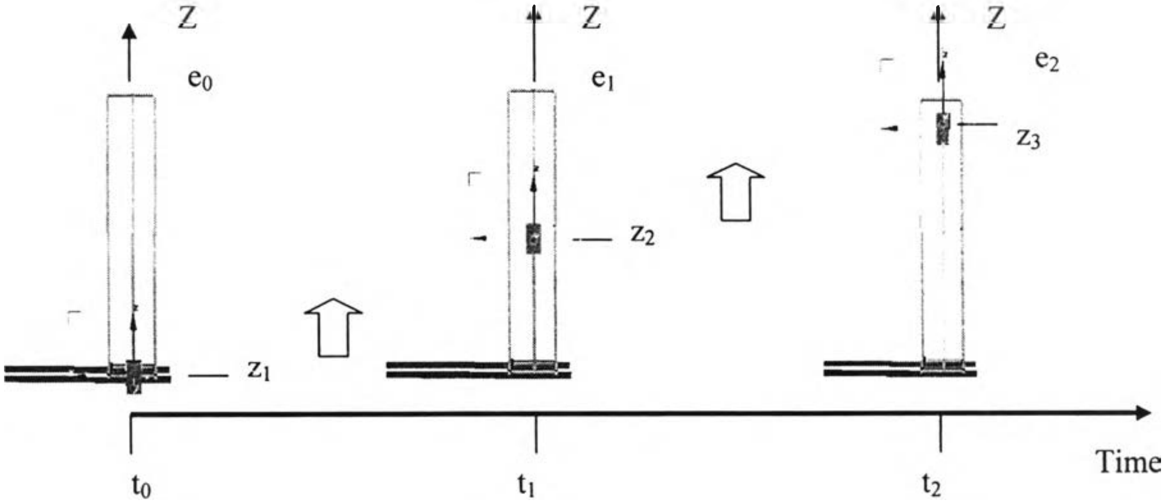


Figure 5.6: Movement of pile-driving hammer along the Z-axis controlled by a position parameter

2) Rotation Parameter

The rotation parameter is the parameter used for controlling the object's rotation in 3D space, e.g., the rotation around the X-axis, Y-axis and Z-axis.

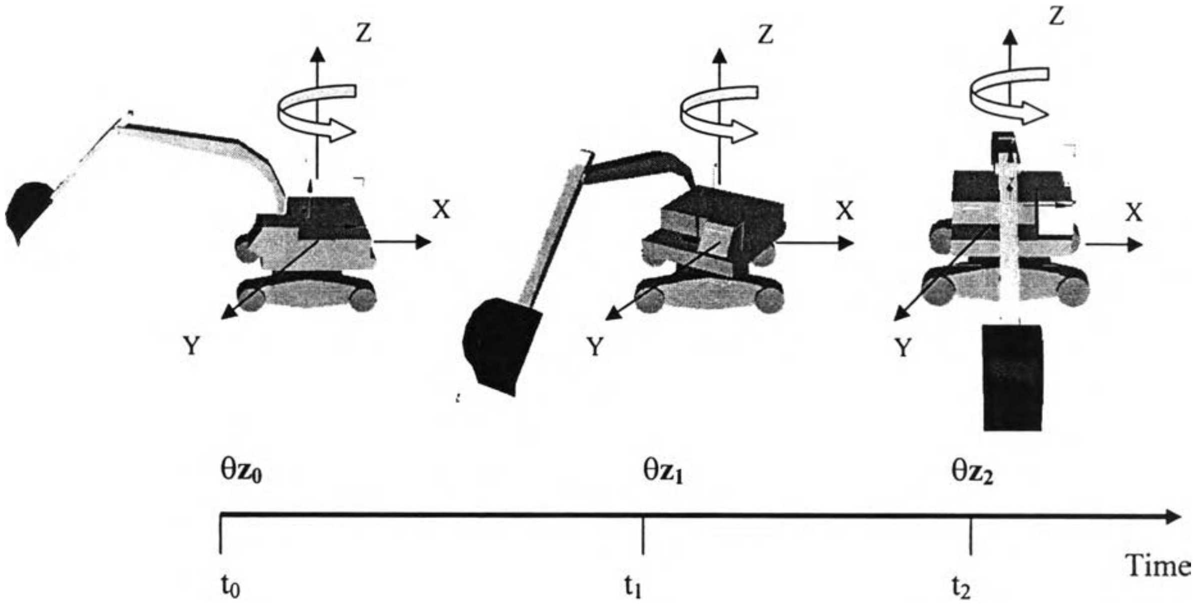


Figure 5.7: The rotation of excavator the body around the Z-axis controlled by a rotation parameter

In Figure 5.7, at time t = t_0, the angle of the excavator body was θz_0 around the Z-axis. When the time changed to t_1, the angle of the excavator body changed around the Z-axis to θz_1 and when the time changed to t_2, the angle of the excavator body changed around Z-axis to θz_2, respectively

3) Shape Transformation Parameter

The Shape transformation parameter is the parameter used for transforming the object's shape in 3D space, e.g., the object's shape transforming in the direction of X-axis, Y-axis and Z-axis.

Figure 5.8 represents the length of pile that was reduced along the Z-axis direction from L_{z0} to L_{z1}, L_{z1} to L_{z2} and L_{z2} to L_{z3} when the time changed from t_0 to t_1, t_1 to t_2, and t_2 to t_3, respectively.

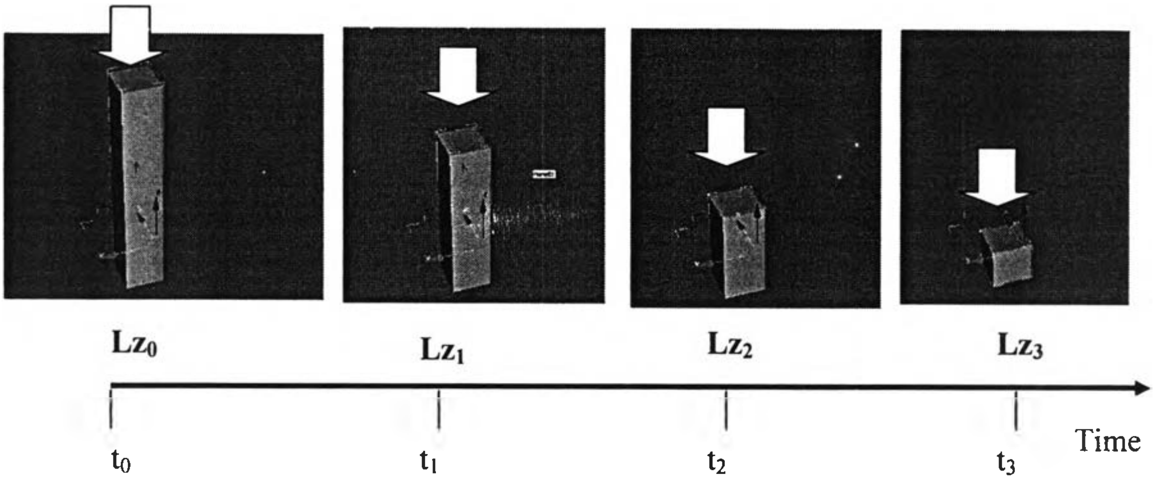


Figure 5.8: The length transforming of pile along Z-axis controlled by a shape transformation parameter

5.2.2.2 Linking Parameters

The link parameter controls the transferring of hierarchical links from one object to another. The link controller respects the link inheritance settings applied to the child object. The object using a link controller is not a true child object. It does not appear in the sub-tree of any linked parent objects.

In Figure 5.9, at time t = t_0, the truss was controlled by the excavator's bucket. When the time changed to t_1, control of the truss was changed from the excavator's bucket to the truck-crane's hook. The truss was linked to be a child of the hook instead. The movement of the truss is controlled by the mobile-crane at time t_2.

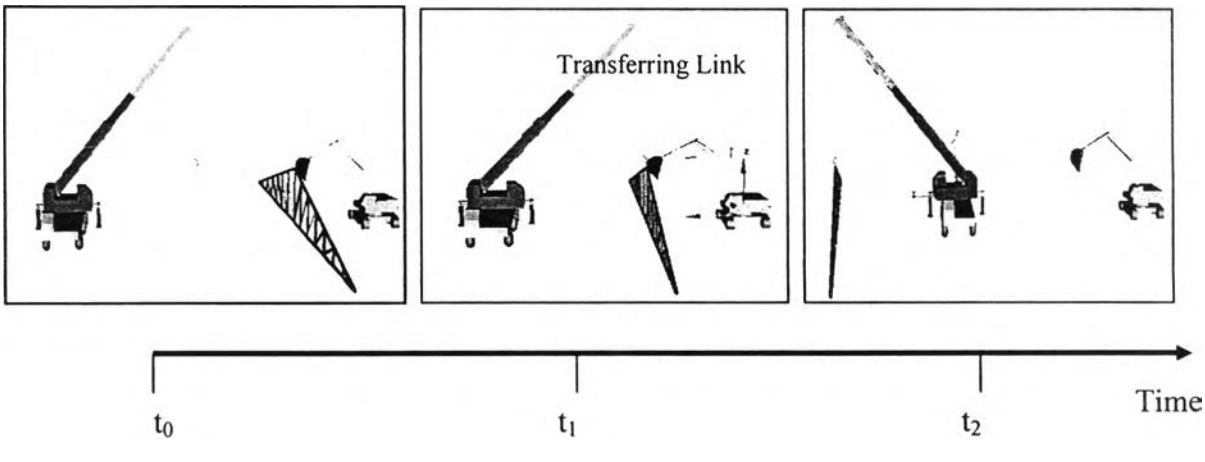


Figure 5.9: The transfer of a hierarchical link from the excavator to the mobile-crane.

5.2.2.3 Visibility Parameters

The visibility parameter is the parameter for controlling the visible of the object in 3D space. It is used to illustrate the appearance of the object according to time changed.

Figure 5.10 presents an application of the visibility parameter to illustrate the earth excavation. At time t = t_0, the excavated soil was visible and the heap of soil was in visible. When the soil was excavated at time t = t_1, the excavated soil was no longer visible but the heap of soil was visible instead.

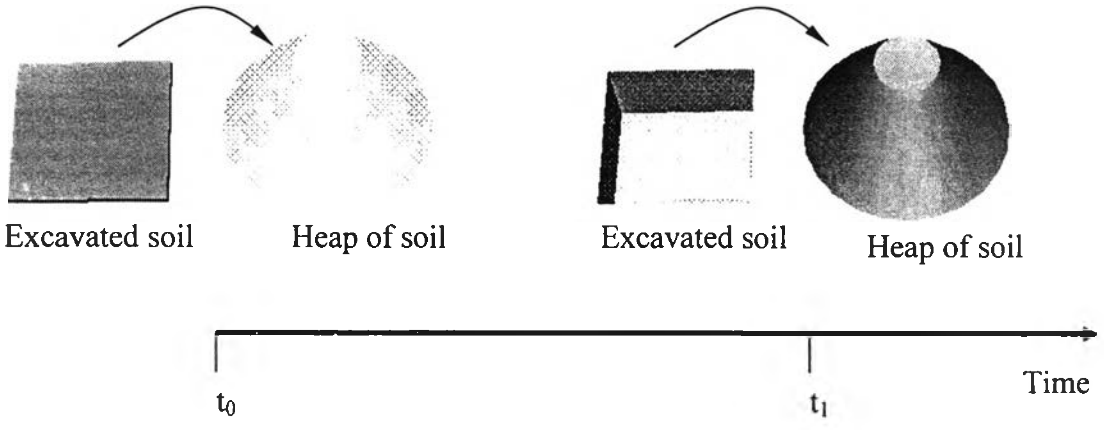


Figure 5.10: An application of the visibility parameter to illustrate earth excavation.

5.2.3 Frame Updating Technique

5.2.3.1 Key-Frames

For this system, key-frames record only the beginning and the end of each transformation of an object or element in the scene. The values at these key-frames are called *keys*. In Figure 5.11, key no.1 was recorded as the beginning of excavator rotation and key no. 2 was recorded as the end of the excavator rotation. The frames that occur between key no.1 and key no.2 are virtual frames that are able to be visualized.

5.2.3.2 Frame Rate

The frame rate of an animation is generally expressed in frames per second (fps). Specifically, this is the number of frames displayed for every second of the real time. Different recording devices lead to different frame rates.

The standard frame rates are as follows:

Frame rate of **NTSC** video = 30 frames per second

Frame rate of **PAL** video = 25 frames per second
Frame rate of **Film** = 24 frames per second

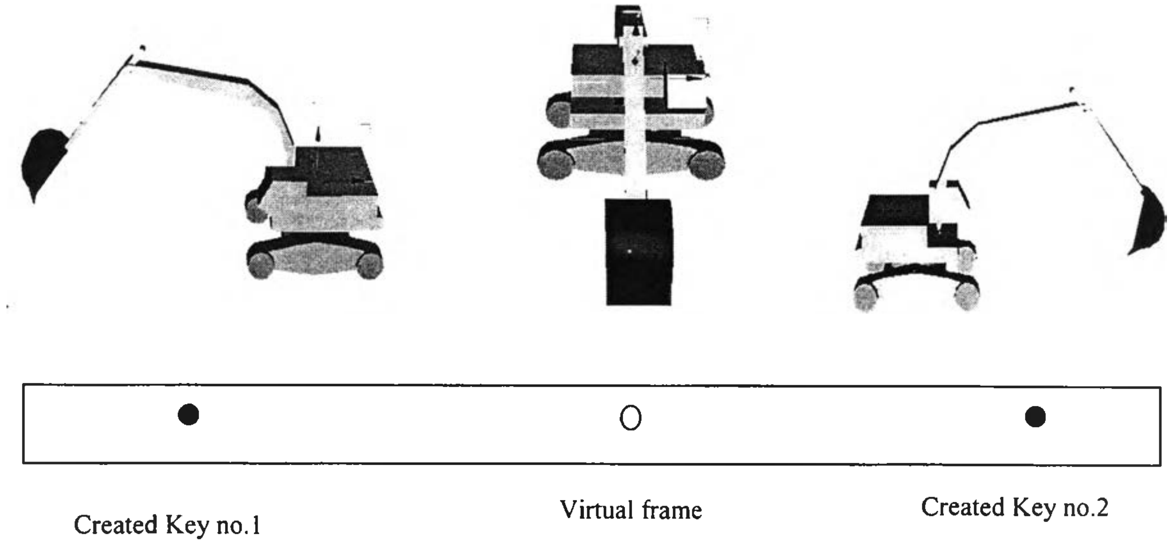


Figure 5.11: Recording of key-frames

The frame rate can be changed for different output at any time, causing the correct number of frames to maintain the correct playback speed for different animation.

For example, if the users create a 30-frame animation for video using an NTSC frame rate of 30 frames per second, the result will be three seconds of animation as show in the following calculation.

Number of animation frames = 30
NTSC frame rate = 30 frame per second
Thus, the animation time = 30/30 = 1 second

On the other hand, if the creators need animation in PAL video (at 25 fps), the frame rate can be switched to the PAL frame rate. The rate of 30 fps is converted to 25 fps, producing the same total animation time with a different number of frames.

Figure 5.12 describes the frame rate of the excavator moving. The speed of this excavator is 4.5 km./hrs. or 1.25 meters per second. For this speed, the average time for 100 meters of actual distance is 80 seconds. The 100 meters of actual distance was scaled down to 1.0 animation-unit that requires 1.0 second for animating the excavator moving. If the 10 frames (frame no.1, 2, 3, ..., and 10) were animated within 1.0 second, the frame rate for the animation would be 10 fps (See Figure 5.12a).

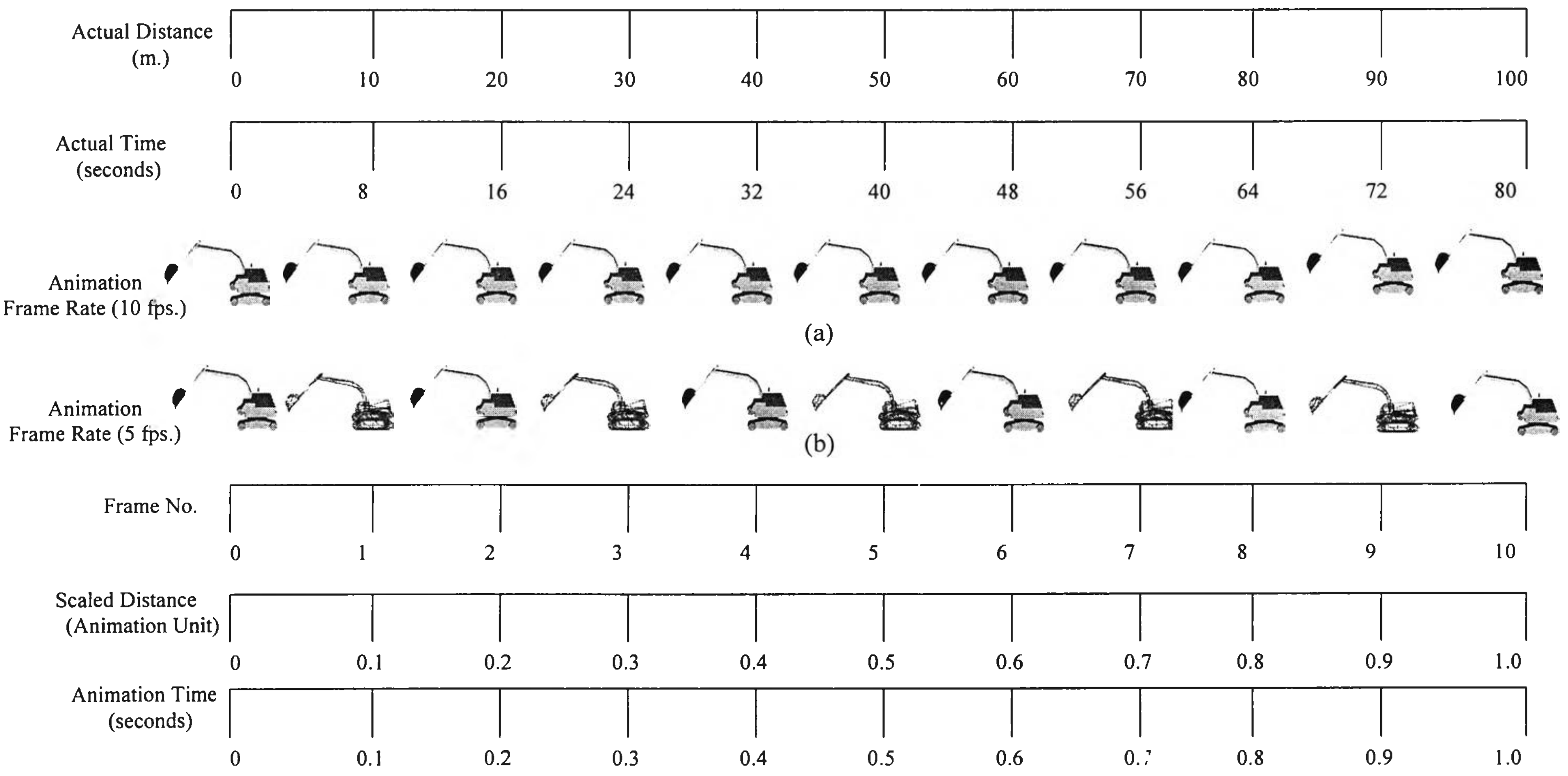


Figure 5.12: Animation frame rate of an excavator moving

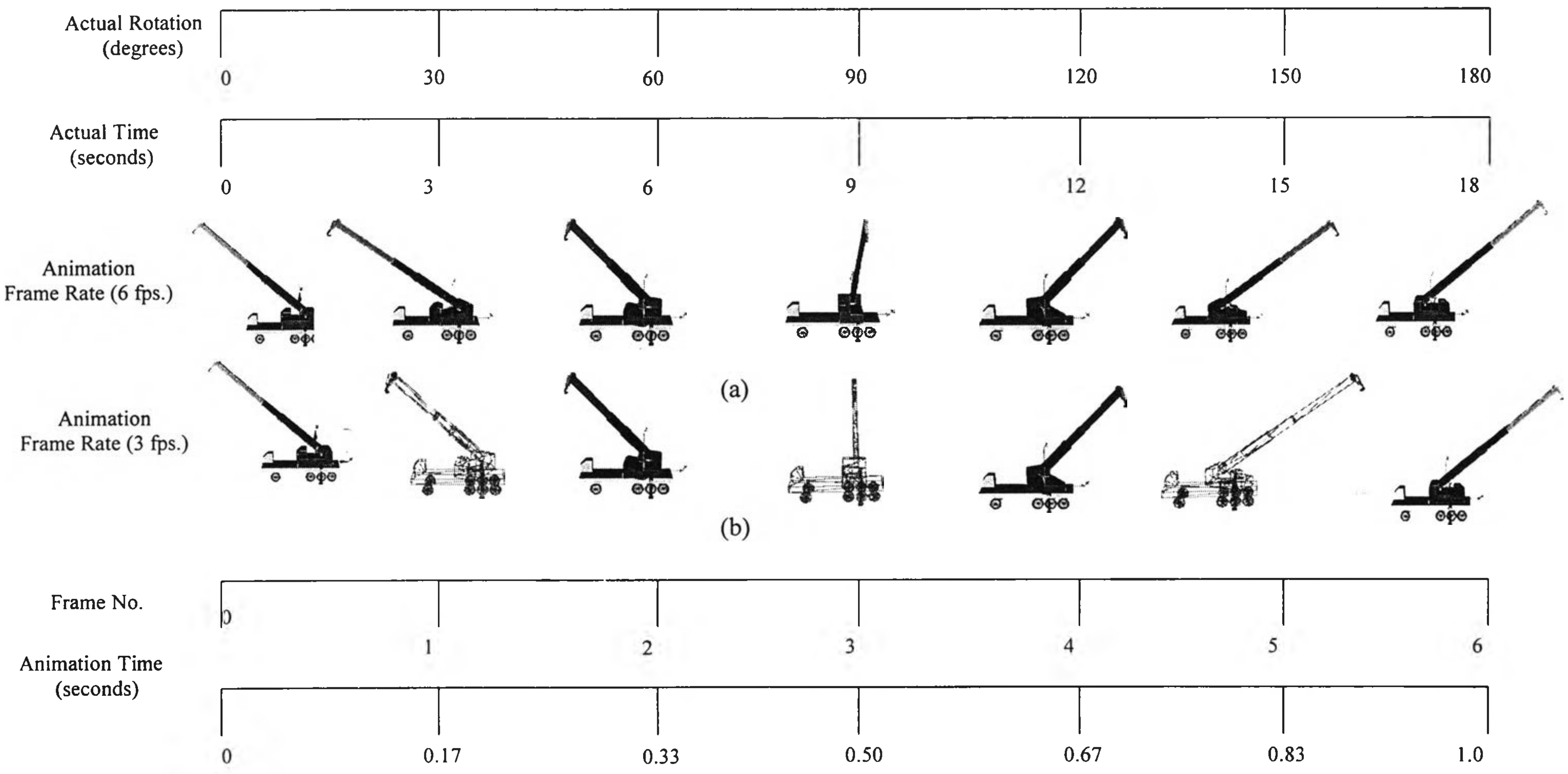


Figure 5.13: Animation frame rate of mobile-crane rotation

However, if five frames (frame no.2, 4, 6, 8 and 10) were animated within 1.0 second, the frame rate of the animation would be 5 fps. (See Figure 5.12b)

Figure 5.13 explains the frame rate of truck-crane rotation. The rotation rate of this crane is 10 degrees per second (rotate180 degrees within 18 seconds of actual time). When the actual time was scaled down to 1.0 second for illustrating the rotation of six frames, the frame rate of the rotation would be 6 fps (See Figure 5.13a).

On the other hand, if only three frames of rotation (frame no. 2, 4, and 6) were displayed, the frame rate of the rotation would be to 3 fps. (See Figure 5.13b)

5.2.4 Visualizing Dynamic Construction Activities

The scope of this research was limited to the study of factory construction only. Thus, the activities that was applied for visualizing the dynamic construction activities were limited to the following factory construction activities:

1) Construction-site preparing
2) Pile driving
3) Pile cutting
4) Earth excavation and backfilling
5) Footing and foundation work
6) RC column work
7) RC beam work
8) Steel base-plate installation
9) Steel column installation
10) Roof structure providing and installation
11) Roof material providing and installation
12) RC slab casting
13) Wall and siding installation
14) Doors and windows installation
15) Temporary works providing

For visualizing the above factory construction activities in 3D space, three essential procedures must be performed:

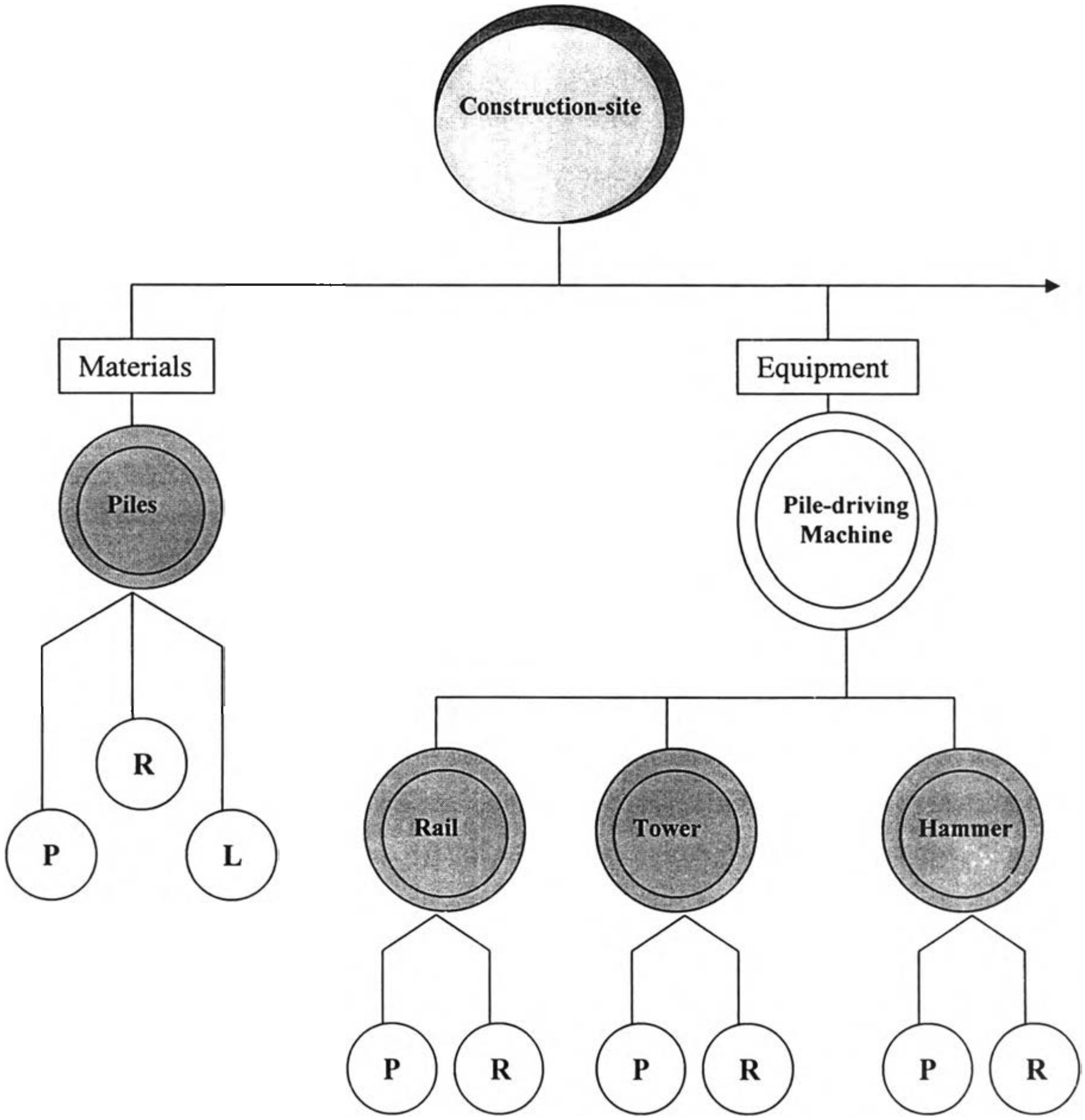
- Set up the resources (materials and equipment) used for operating in each activity (See Table 5.1).
- Assign controlling parameter to each activity (See Table 5.1).
- Create key-frames to control the visualizing of materials and equipment.

For example, the materials and equipment used in pile-driving activity are pile and pile-driving machines, respectively. The controlling parameters for visualizing the movement of piles are position, rotation and linking parameter. The controlling parameters for visualizing pile-driving machine operation are position and rotation, respectively. For the other construction activities, the visualizing resources (materials & equipment) and their controlling parameters are presented in Table 5.1.

Table 5.1: Controlling parameters for visualizing the factory construction activities.

No.	Activity Name	Visualizing Resources		Controlling Parameter
1	Pile driving	**Material:**	Piles	Position, Rotation, Linking
		Equipment:	Pile-driving machine, Trailer	Position, Rotation
2	Pile cutting	**Material:**	Piles	Shape Transformation
		Equipment:	-	-
3	Earth excavation and backfill	**Material:**	Soil	Visibility
		Equipment:	Excavator	Position, Rotation
4	Footing and Foundation casting	**Material:**	Concrete, Formwork	Visibility
		Equipment:	Concrete mobile-truck	Position, Rotation
5	RC column casting	**Material:**	Concrete, Formwork	Visibility
		Equipment:	Concrete mobile truck	Position, Rotation
6	RC beam casting	**Material:**	Concrete, Formwork	Visibility
		Equipment:	Concrete mobile truck	Position, Rotation
7	Steel base-plate installation	**Material:**	Steel base-plate	Visibility
		Equipment:	-	
8	Steel column installation	**Material:**	Steel column	Position, Rotation, Linking
		Equipment:	Truck-crane	Position, Rotation
9	Roof structure providing and installation	**Material:**	Roof Truss, Purlin	Position, Rotation, Linking
		Equipment:	Truck-crane, Excavator	Position, Rotation
10	Roof material providing and installation	**Material:**	Roofing material	Visibility
		Equipment:	Truck-crane	Position, Rotation
11	RC wall casting	**Material:**	Concrete, Formwork	Visibility
		Equipment:	Concrete mobile truck	Position, Rotation
12	RC slab casting	**Material:**	Concrete, Formwork	Visibility
		Equipment:	Concrete mobile truck, concrete pump, dump truck, grader, Roller	Position, Rotation
13	Wall and siding installation	**Material:**	Wall and siding material	Visibility
		Equipment:	-	-
14	Doors and windows installation	**Material:**	Doors and windows	Visibility
		Equipment:	-	-
15	Temporary works providing	**Material:**	Temporary material	-
		Equipment:	-	
16	Construction Site preparing	**Material:**	Fence, Site office, Material store, Labor camp	Visibility

For ease of understanding, the graphical structures of the visualizing resources and the controlling parameter of pile driving are illustrated in Figure 5.14. The graphical structures of the visualizing resources and controlling parameters of the other activities of factory-construction are presented in appendix 4.



P = Position Parameter
R = Rotation Parameter
L = Linking Parameter

Figure 5.14: Graphical structure of visualizing pile-driving activity

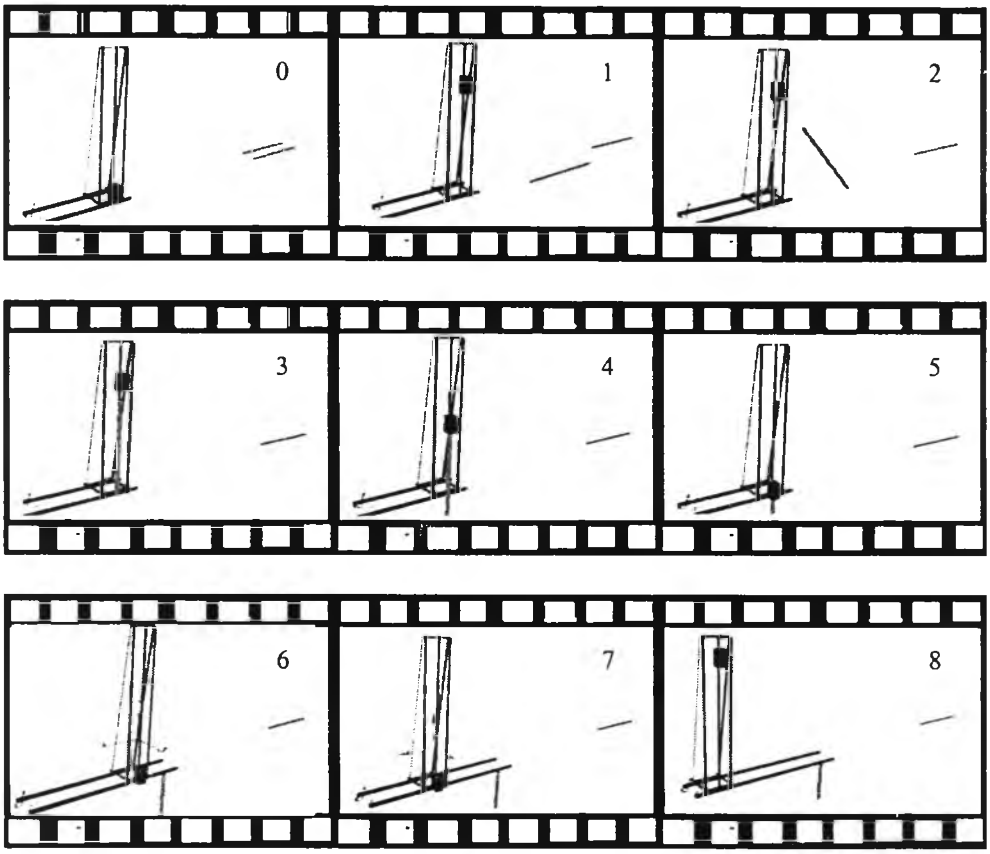


Figure 5.15: Key-frames of pile-driving visualization

Figure 5.15 presents the key-frames created for visualizing pile-driving activity. The following key-frames are explained as in the following details:

1) At key-frame no.0, first pile and pile-driving machine are ready to be operated.

2) At key-frame no.1, the hammer of the pile-driving machine has been moved along the Z-axis by using the position parameter. The first pile has been moved along the X-axis using the position parameter.

3) At frame no.2, the first pile has been rotated around the Y-axis using the rotation parameter.

4) At frame no.3, the first pile has been rotated to be in a vertical position using the rotation parameter, and is ready to be driven.

5) At frame no.4, the first pile has been driven by the hammer of the pile-driving machine, which was controlled by the position parameter.

6) At frame no.5, the first pile has been completely driven.

7) From frame no.6, to frame no.8, the tower of the pile-driving machine has been moved along the X-axis to the next position, which has been controlled by the position parameter.

5.3 Simulation Techniques

Up to now, simulation has been one method applied to solve complicated problems. Currently, computer technology has been developed to be able to solve more complicated problems. In construction operations, complex problems need to determine such as construction techniques, methods and sequences, quantities of equipment and operation costs. Thus, the computer technology can be applied to develop the simulation models to assist the planners in their decision-making. This chapter describes the basic principle of simulation processing and the method for generating a simulation model. The simulation models of factory construction, which was the scope of this research are developed by using this basic methodology.

5.3.1 Discrete Event Simulation

In a real dynamic system or dynamic process, there is at least one event that occurs at a point of time. Simulation is a process for modeling the real dynamic systems. Discrete event simulation is defined on the assumption that " the state of the system was instantaneously changed by event occurrence at a specific time".

Law (2000) explained that discrete event simulation is the modeling of a system that evolved over time by a representation in which the state variables change instantaneously at separate points in time or change at only a countable number of points in time, where an *event* is defined as an instantaneous occurrence that changes the state of the system.

Martinez (1996) explained how discrete event simulation can be performed on a computer through the use of general purpose programming languages or specific simulation tools.

5.3.2 Time Mechanism

The nature of a discrete event simulation model is dynamic. Thus, the simulation models need a mechanism to advance simulated time from one value to another. The "*Simulation clock*" is the variable in a simulation model that generates the current value of simulation time. The simulation clock is not the time for running the simulation model on the computer. Generally, there is no relationship between time for running the simulation model and simulation clock. The principle for advancing the simulation clock is divided into two approaches: next-event time advance and fixed-increment time advance. For generating time advance in discrete- event simulation model using generalpropose programming, the approach of fixed-increment time advance is easier than the approach of next-event time advance. But in the opinion of most simulation-modelers, the time advance generated by the next-event advance approach is more effective than the fixed-increment time advance approach due to the fact that time and event occurrence will be in harmony if the appropriate fixed- time is selected. In general, the appropriated fixed-time is often a small value that can require a lot of computer time for operating the simulation model.

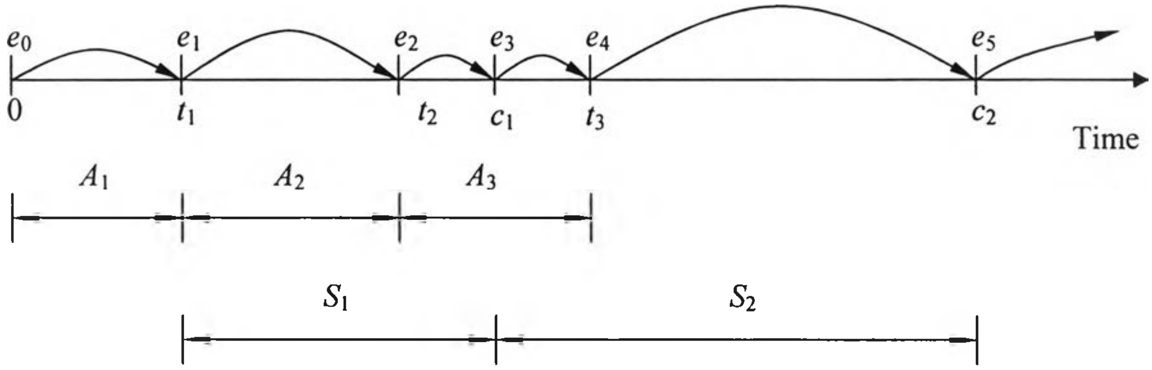


Figure 5.16: Next-event time advance approach illustrated for a single-server queuing system (Source: Law, 2000).

Figure 5.16 illustrates the next-event time advance for a single-server queuing system, where the following notations are used:

t_i = time of arrival of the i^{th} customer ($t_0 = 0$)

A_i = interval time between the i^{th} and $(i-1)^{th}$ arrival of customer = $t_i - t_{(i-1)}$

S_i = time that server actually spends serving the i^{th} customer
(Exclusive of customer's delay in queue)

D_i = delay in queue of the i^{th} customer

c_i = $t_i + D_i + S_i$ = time that i^{th} customer complete service and departs

e_i = time of occurrence of the i^{th} event of any type
(The i^{th} value the simulation clock take on, excluding the value $e_0 = 0$)

According to Law (2000), two disadvantages of fixed-increment time advance are: 1) the errors introduced by processing events at the end of the interval in which they occur and 2) the necessity of deciding which event to process first when events that are not simultaneous in reality are treated as such by the model. These problems can be made less severe by making Δt smaller, but this increases the amount of checking for event occurrences that must be completed and results in an increase of execution time. Because of these problems, fixed-increment time advance is generally not used for discrete-event simulation models. Figure 5.17 illustrates the fixed-increment time advance approach.

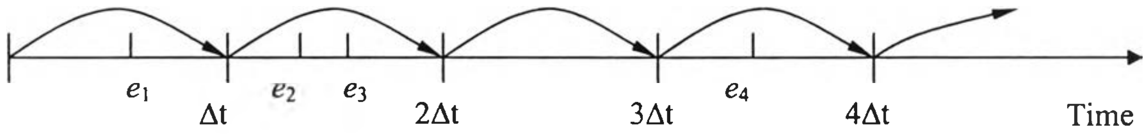


Figure 5.17: Fixed-increment time advance approach (Source: Law, 2000)

In this approach, the simulation clock is advanced in increments of exactly Δt time units for some appropriate choice of Δt. After each update of the clock, a check is made to determine if any events should have occurred during the previous interval of length Δt. If one or more events are scheduled to occur during this interval, these events are considered to occur at the end of the interval, and the system state is then updated.

5.3.3 Monte Carlo Simulation Technique

In general, the data used as input in simulation models for solving complex problems is usually a quantity or numerical value. However, modelers cannot determine what values are chosen as the input value of the model that can simulate real world problems. The answer is those numerical data must be randomized numbers in order to be able to simulate real world problems. The popular technique usually used for modeling real world problems by randomizing data is called the "Monte Carlo" simulation technique.

The Monte Carlo technique is a technique used to generate simulation data by using random numbers and accumulated probability (Tongprasert, 2001). These random numbers can be generated by several methods. For example, they can be obtained from random number tables, generated by using dice or wheel, or generated by computer programming. Those methods can generate random numbers to be a uniform distribution. Accumulated probability values of requirement data come from data records or by trials.

The Monte Carlo technique is performed by using the following steps (adapted from Tongprasert, 2001).

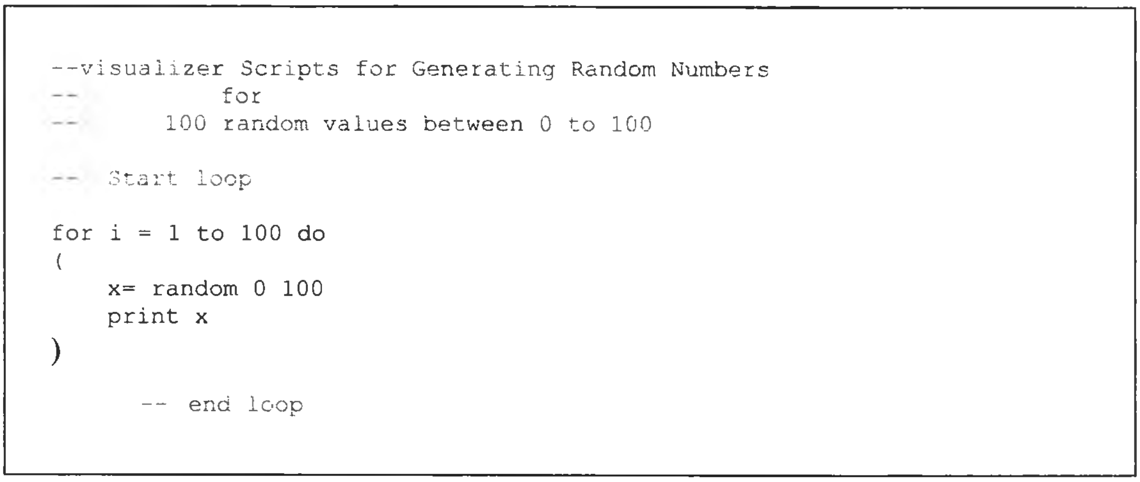
1) Establish graphs or tables of the accumulated probability of requirement data.
2) Generate random numbers by using computer programming (visualizer Script) and convert those values to be 0 to 1.

3) Use the converted values in step 2 instead of the accumulated probability data.
4) Read the values of requirement data from graph or table when the accumulated probability equals the value in step 3
5) Repeat performing in step 2 to 4 until there are enough amounts of data requirement

In this research, computer programming was selected as a tool to generate random numbers. Historical data recorded from construction operations was chosen for the accumulated probability values. Kinds of such historical data used include productivity data of pile driving, steel-columns installation and roof-trusses installation, as shown in Appendix 7.

5.3.4 Validating Random Numbers

In this research, the properties of random numbers are classified into two categories: *uniformity* and *independence*. In order to validate the ability of computer programming to generate random numbers, these numbers should be tested. In this research, both uniformity and independent properties of random numbers are tested by following using statistical methods: (Source: Tongprasert, 2001)



```
--visualizer Scripts for Generating Random Numbers
--          for
--      100 random values between 0 to 100

--  Start loop

for i = 1 to 100 do
(
    x= random 0 100
    print x
)

      -- end loop
```

Figure 5.18: Visualizer scripts for generating random numbers.

1) The uniformity is validated by using the method of *Frequency Test.*
2) The independence is validated by statistical methods as follows:
    2.1) *Runs Test*
    2.2) *Gap Test*
    2.3) *Autocorrelation Test*
    2.4) *Poker Test*

This research employed computer software called "3DS Visualizer" as a tool for simulating building construction and applied Visualizer Script language for programming construction operation. Thus, the random numbers generated by Visualizer Script should have their random properties validated. In order to validate the random properties of random numbers generated by computer, the Visualizer scripts are used to generate ten sets of one hundred random numbers as shown in Figure 5.18. Each set of random numbers was tested and validated by the above testing methods as follows:

5.3.5.1 Frequency Test

The frequency test is used for validating the uniformity of probability distribution. In this research, two statistical methods are used as tools in the frequency test. These methods are the Chi-square Test and Kolmogorov-Smirnov. Null hypothesis of the testing is "the distribution of random numbers is uniform distribution"

5.3.5.2 Running Test

The running test is used for validating the running of random numbers by considering the run-up and run-down of numbers. The order of the numbers should not be constant. Thus, the order of random numbers should not be circle or trend.

5.3.5.3 Gap Test

The gap test is used for checking the frequency of the numbers that appears in the series of random numbers in comparison to the frequency of expected numbers.

5.3.5.4 Autocorrelation Test

The independent random numbers should not be correlated. Thus, the autocorrelation test is used to test the independence of random numbers. This method is used for testing the affectation of the former generated number to other random numbers.

5.3.5.5 Poker Test

The poker test is a comparison between the frequency of random numbers that have different and same positions with the expected frequency when random numbers are independent.

The rejection or acceptance of null hypothesis within α value can be converted to percent confidence of the uniformity and independence by the following equation.

$$\text{Percent confidence} = (1 - \alpha) \times 100 \tag{8.1}$$

5.4 Construction Data

The Monte Carlo technique requires two types of data; e.g., 1) random numbers and 2) accumulated probability of requirement data. In this research, random numbers are generated by computer programming (Visualizer Scripts). Note that the accumulated probability of requirement data refers to accumulated probability of building construction data that are related to the construction productivity. These data were collected from historical field records and recorded form construction sites.

Construction project data is an important element of for project planning. Thus, in order to achieve a simulation of the factory construction process using Integrated System, which is developed in this research, a comprehensive database was designed and implemented. Construction project data and information were collected or captured from various sources such as designers, main contractors, sub-contractors, equipment rental suppliers and material suppliers. In order to design the Integrated System database, the collected data must be carefully identified, classified and analyzed.

5.4.1 Construction Data Types

In the construction process, the construction data that is usually related to construction time and productivity can be classified according to their properties into two categories: 1) the numerical data, and 2) the physical data. In this research, the numerical data refers to the data that can be calculated. The numerical data that are involved in construction time are the construction productivity data and quantity of resources used; e.g., the productivity of pile driving by one driving machine, the productivity of pile cutting by two gangs of labor and the productivity of earth excavating by one excavator. On the other hand, physical data cannot be calculated; e.g., construction techniques, methods, sequence and type of machine used for building-structure installation. The data that related to construction cost are material prices, temporary-work prices and price rates of construction resources; e.g., the price rate for renting construction machines and equipment, the price rate of manpower and labor, and the price rate for renting facilities. All of these data can be explained as shown in Figure 5.19.

In the Figure 5.19, in order to simulate building-construction process, time and cost, the Integrated System requires both categories of construction data. The numerical data related to productivity and quantity of resources is used as an input of the Monte Carlo technique to determine construction time. The physical data concerning construction technique and resources are used for setting the construction process and resources used by visualizing the construction process, and construction cost can be determined by using the price rate of construction resources.

5.4.2 Examples of Construction Data

Building construction data used to simulate construction time are usually collected from historical records and recorded from construction sites. An example of building construction data involved determining construction time is the productivity of pile driving recorded from construction site. The pile driving data should be the main factor affecting the operation time of pile driving such as the type or sections of piles, the length of the piles and properties of pile-driving machines as shown in Table 5.2. Other examples of building construction data are the productivity of pile cutting, the productivity of earth excavating by excavator and the productivity of roof-structure installation.

Table 5.2: Productivity of pile driving by one pile-driving machine.

PILES DRIVING DATA			
PILE NO.		**PILE LENGTH (M.)**	**DRIVING TIME (MIN.)**
I 26	**I 30**		
203		16.00	18
204		16.00	23
205		16.00	20
	139	16.00	25
166		16.00	23
165		16.00	30
207		16.00	31
206		16.00	28
	208	16.00	24
	251	16.00	51
	253	16.00	30
119		16.00	17
118		16.00	20

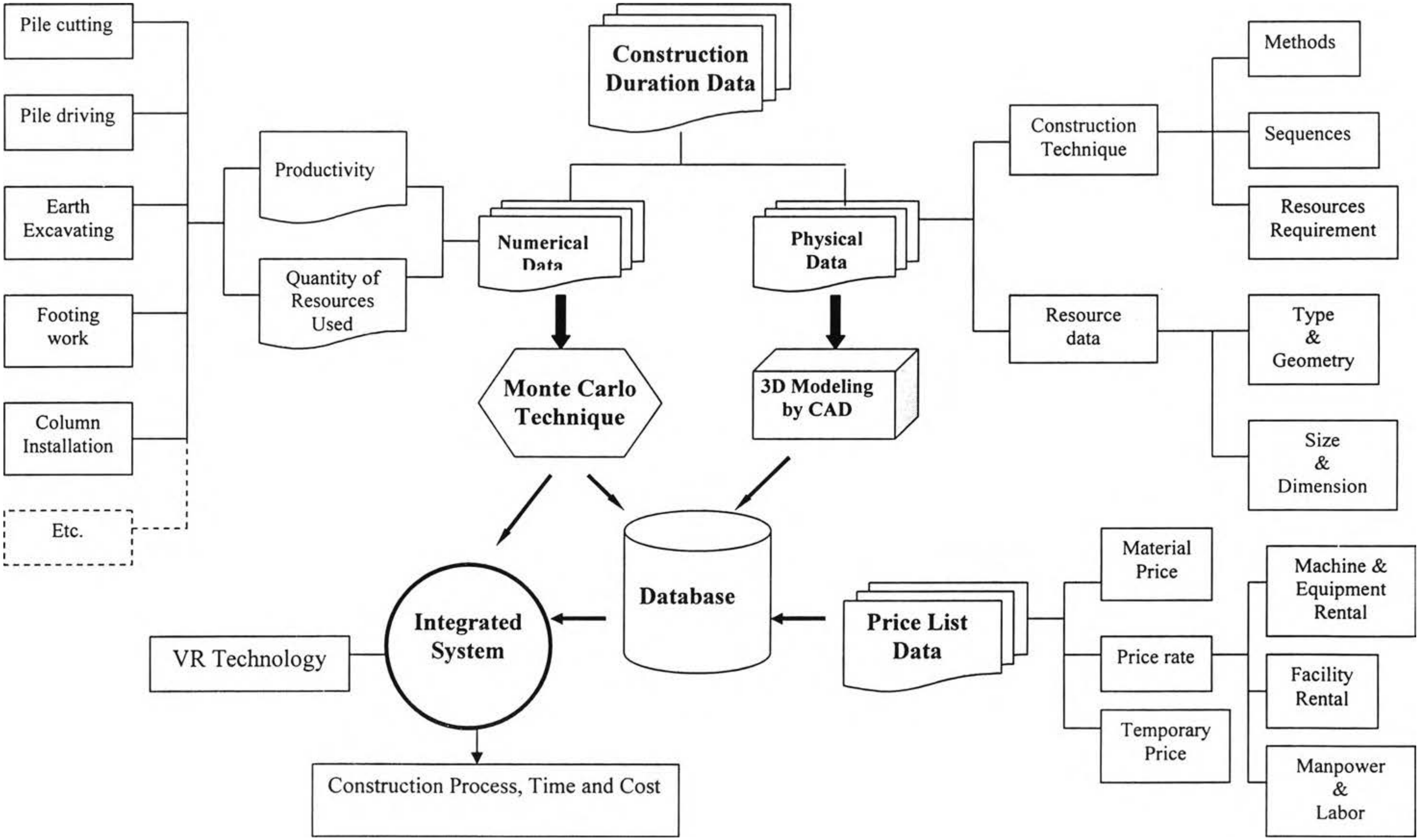


Figure 5.19: Building construction data used to simulate construction process, time, and cost

5.5 System Database

In the database development process, the database designer should clearly define database structure that consists of database storage files and program control files as presented in Figure 5.20. According to Kaewkangwan (1993), database development processes are divided into two main processes: 1) database definition and 2) input source data.

Based on the above literature, the methods for developing an Integrated System database in this research are divided into five processes as follows.

1) Data classification
2) Data identification
3) Data acquisition and collection
4) Data analysis
5) Database development and implementation

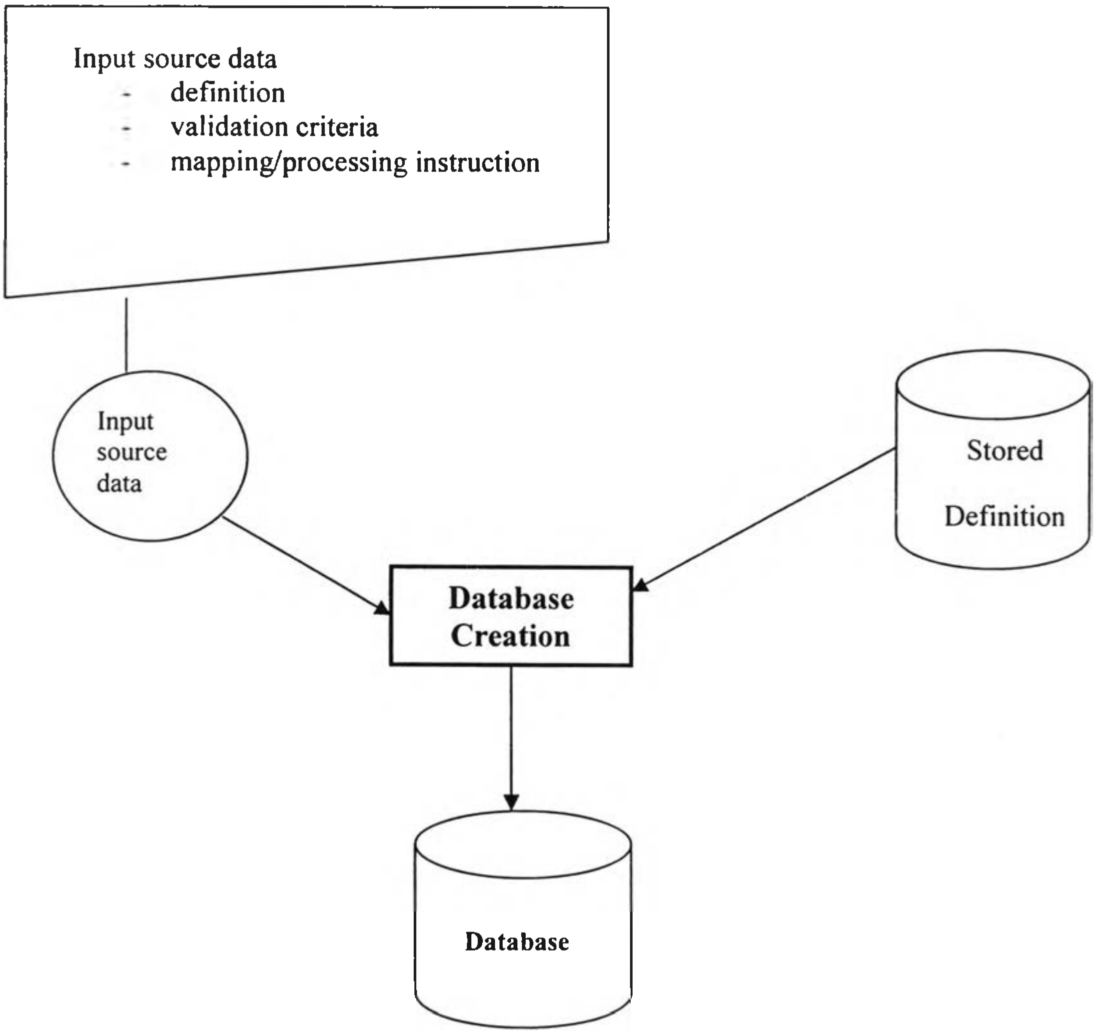


Figure 5.20: Database development processing (Source: Kaewkangwan, 1993)

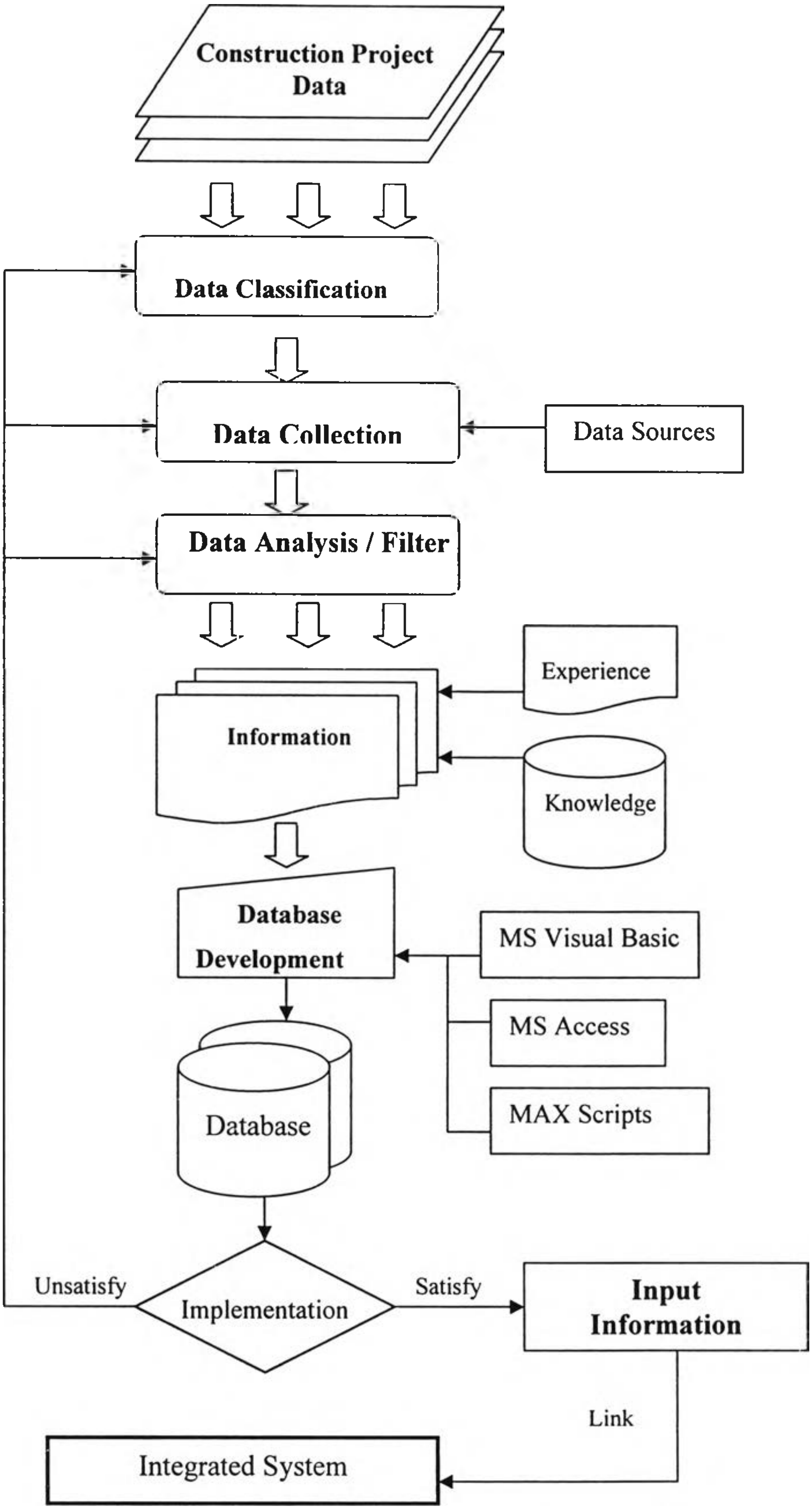


Figure 5.21: Flow chart of Integrated System database development.

Data from many data-sources was identified and filtered to obtain only the essential data for the Integrated System requirement. For example, the Integrated System requires the data about the excavator to simulate the excavating work. The equipment data of the excavator from the handbook consists of various parts. Thus, these data should be identified and filtered to obtain only the important parts such as type, size, dimension, travel speed and productivity. Size and dimension figures are used for creating the 3D model. Speed and productivity are used for calculating the operation time.

After the data were identified and filtered, the next step was to classify the identified data into appropriate data groups. The essential data for the Integrated System was classified into main groups that can be easily collected and analyzed. After the system data was classified in the step of the data collection, the researcher could explore the expected data from many data sources by following the data groups that were classified. The collected data had to be analyzed in many ways; e.g., interviewing experts and brain storming to neglect missing data, and testing the data accuracy using statistical tools. After the collected data were already analyzed by several methods, it became the information that was ready to be used to develop the Integrated System database. The software tool called "MS Visual Basic" was used as a tool for developing the database and for implementation. It can be linked together with the Virtual Reality (VR) software that is an important part of the Integrated System. Figure 5.21 is a flow chart that describes the process of database development.

5.5.1 Data Classification

There are various data from different sources used in a construction project. In order to establish the Integrated System database, construction data from many sources had to be classified into the same type according to the physical properties and construction planning processes. The data were classified into two types by their physical properties such as graphical data, and non-graphical data, and were classified into four types by construction planning process, such as building-component, resource, facility and construction technique data as shown in Figure 5.22.

5.5.2 Identification of Data Properties and Details

Before the classified data is collected and acquired, their properties and details must be identified in order to distinguish what properties and details of the data should be collected and recorded. In the previous step, the system data were classified according to the construction planning process into four groups. The following system data were described about properties and details.

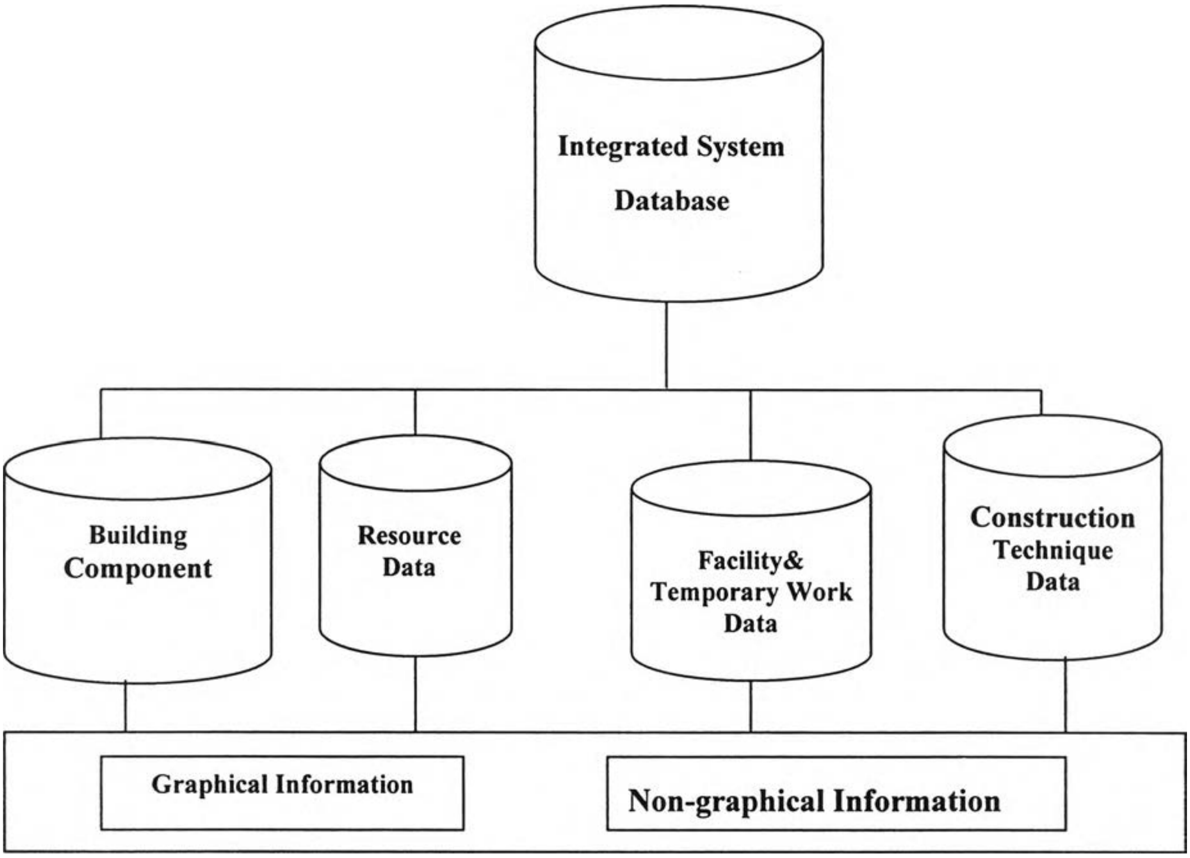


Figure 5.22: Structure of Integrated System data classification

5.5.2.1 Building-Components Data

The Integrated System needs building-component data to distinguish the quantity of construction work that will be simulated by the system since the quantity of construction work can affect construction duration and cost. In this research, the building-component data were identified by using Work Breakdown Structure (WBS). In general, building construction work can be broken down according to construction sequences and building-components into the following works: 1) piling work, 2) excavation work, 3) pile cutting work, 4) foundation work (footing and ground-beam), 5) RC column work, 6) steel base-plate installation work (in case of factory-building), 7) steel column work, 8) roof-structure work, 9) RC slab work, 10) wall and siding work, and 11) doors and windows work.

Each of the building construction work consists of different building-components and each building-component consists of different properties and details. Moreover, the data about building-component properties and details comes from many sources. Therefore, the properties and details of the building-components should be identified that can lead the researcher to know the correct data sources.

The properties of the building-component data in this research are divided into two categories: physical properties and construction system properties. The building-component data are broken down according to physical properties into geometry, size and dimension, color, weight and quantity, and are also used to identify the construction system based on design criteria. Examples of construction system types are driven-pile or bored-pile, RC-column or steel-column, and roof-truss or roof-frame. Types of construction systems will identify and relate to the construction method and performance. Table 5.4 and Figure 1 in Appendix 5 clearly illustrate the properties and details of the building-component data.

5.5.2.2 Resource Data

Resource data are important data for an Integrated System used to simulate the construction process, duration and cost. In general, construction resources consist of material, manpower and equipment. In the previous topic, the material has been already identified properties and details by building-component data. Therefore, properties and details of resource data remain to be identified only construction equipment and manpower. In this research, construction equipment consists not only of construction machines but also of temporary works. Properties and details of construction machine data are identified as follows: 1) types, 2) size & dimension, 3) productivity, 4) performance, 5) cost, and 6) working-space requirement. Properties and details of manpower consist of: 1) types, 2) gang size, 3) productivity, and 4) cost. All of these were presented in Table 5.3 and Figure 2 in Appendix 5.

5.5.2.3 Facilities Data

In construction-site planning, construction facilities were classified into two categories: existing facilities on site and facilities located on the construction site. Examples of existing facilities on site are existing-infrastructures such as electrical pipelines and water supply pipes. The facilities located on the site refer to construction-site facilities, which were used to support the construction operation, such as site-office, material-storage area, prefabricated area, fence, and labor camp. Examples of temporary work are scaffolding, formwork, and column bracing. Temporary works are described in properties and details as follows: 1) types; 2) size & dimension; 3) cost; and 4) space requirements. Properties and details of existing facilities on site are identified as follows: 1) types; 2) size & dimension; and 3) position. Properties and details of facilities located on site are 1) types; 2) size & dimension; 3) cost; 6) working-space requirements. Table 5.5 and Figures 3 in Appendix 5 show properties and details of the facility data.

Table 5.3: Properties and details of resource data

No.	**Resources**	**Properties and Details**					
		Type	**Size & Dimension**	**Productivity**	**Performance**	**Cost**	**Work-space**
1.	Construction Machines	x	x	x	x	x	x
2.	Manpower	x		x		x	

Table 5.4: Properties and details of building-component data.

No.	Building-component	Physical Property						Construction System
		Size	Geometry	Dimension	Color	Weight	Quantity	
1.	Pile	x	x	x		x	x	Driven-pile
2.	Excavated work			x			x	
3.	Pile cutting	x		x			x	
4	RC foundation	x	x	x			x	Cast in place
5	RC column	x	x	x			x	Cast in place
6	Steel base-plate	x	x	x			x	Prefabricate
7	Steel column	x	x	x	x	x	x	Prefabricate
8	Roof structure	x	x	x	x	x	x	Prefab truss
9	Roofing	x	x	x	x		x	Metal sheet
10	Wall & siding	x	x	x	x		x	
11	Doors & windows	x	x	x			x	

Table 5.5: Properties and details of the facilities and temporary work data

No.	Facility	Properties and details				
		Type	Size & Dimension	Position	Cost	Space-requirement
1.	Existing facility on site	x	x	x		
2.	Facility located on site	x	x		x	x
3.	Temporary work	x	x		x	x

5.5.2.4 Construction Technique Data

Construction techniques and construction methods are important information for construction planning. The construction project team must select the appropriate construction techniques and methods to operate the construction process. In this research, construction techniques and method data refer to 1) the construction system; 2) the assembly sequencing procedure; 3) the logical relationship of assembly or constructability; 4) resource requirement. The details of the construction technique data are explained by Figure 4 in Appendix 5.

5.5.3 Data Acquisition and Collection

In this research, data acquisition and collection involve the following topics: 1) data sources, 2) documented data, 3) data sampling, and 4) storing data. The details in each topic are explained as follows:

5.5.3.1 Data sources

In order to develop the database of the Integrated System, the data that was already classified and identified in the previous steps was collected from various sources according to the types of data. For example the building component data was collected from construction drawings, schedules, bills of quantity (BOQ) and design sheets while resources data was collected from construction-machine handbooks and catalogs, rental suppliers, sub-contractors and field recording. In this research, the data sources were divided into two categories: data from existing documents and data from sampling. Table 5.6 is used to present various data sources based on the various types of data that were classified and identified in the previous steps.

5.5.3.2 Existing Document data

Data from existing documents refers to data from design drawings, construction scheduling, engineering design-sheets, handbooks, manuals, catalogs and historical records. These documents were created by construction firms (designer, constructor or consultant) and also brought from suppliers or sub-contractors.

Table 5.6: Data acquisition methods and sources

Items	**Data Types**	**Properties and details**	**Data Acquisition Methods & Sources**		
			Existing documents	**Interview and questionnaire**	**Sampling Records**
1.	Building Components	Type, WBS	BOQ, scheduling		
		Sizes, Dimensions, Color, Geometry	Drawing		
		Weight	Design sheets		
		Quantity	Drawing, BOQ		
		System	Design sheets	Project team	
2.	Resources 2.1 Machines	Type	Catalogs		
		Sizes, dimensions	Handbooks		
		Productivity	Handbooks		Field records
		Cost	BOQ	Supplier	
		Work-space requirement	Handbooks	Supplier	
	2.2 Temporary Work	Type	Catalogs		
		Sizes, dimensions	Catalogs		
		Space requirements	Catalogs		
		Cost	BOQ, Historical	Supplier	
	2.3 Manpower	Type		Sub-contractor	
		Productivity	Historical records		Field records
		Cost	BOQ, Historical	Sub-contractor	
3.	Facilities 3.1 Facilities On Site	Type	Catalogs, Historical	Supplier	
		Sizes, dimensions	Catalogs, Historical	Supplier	
		Cost	Historical records	Supplier	
		Space requirements	Historical records	Supplier	
	3.2 Existing Facilities	Type, Size			Field records
		Position			Field records
4.	Construction Techniques	System Type	Historical records	Sub-contractor	
		Sequences	Historical records	Sub-contractor	
		Constructability	Historical records	Sub-contractor	
		Resource requirements	Historical records	Sub-contractor	Field records
		Cost, productivity	Historical records		Field records
		Space requirements	Historical records	Sub-contractor	

5.5.3.3 Data Sampling

For acquiring and collecting the data, the sampling of data comes from two sources: questionnaire and historical records. The questionnaire is done by either direct interview or questionnaire. The direct interview is done by interviewing construction-site managers or project managers. Questionnaire forms were distributed to construction project to fill out the data and then sent back to the researcher. The historical records in this research refer to data recorded from the field or construction site.

- Questionnaire

The questionnaire consists of many major questions. Each question requires the respondent to choose one or more answers from multiple choices, and fill in the blank spaces. The respondent can also write down more details or more comments. The questionnaire was sent to construction projects and after the project manager or project team had completely answered the questions in the questionnaire, it was sent back to the researcher according to the researcher's address. The questionnaire forms are presented in Appendix 6.

- Historical data records

Historical data is data recorded from actual work. It was recorded from the field or construction site. Examples of historical data are the productivity of pile-driving machines and the productivity of mobile crane for installing roof-structures.

5.5.3.4 Data Storing

The data from existing documents, questionnaires and field records is stored in hard copy form (see Appendix 6). All of these data were then transferred or changed to electronic data. The graphical data from existing documents or catalogs, such as type, size, dimension, geometry and color, were transferred to the CAD data by creating 3D CAD drawing files (*.dwg) (See Appendix 1 & 2). Spreadsheet software was used for storing field data records such as the productivity data of construction machines, manpower and construction methods (See Appendix 7).

5.5.4 Database development

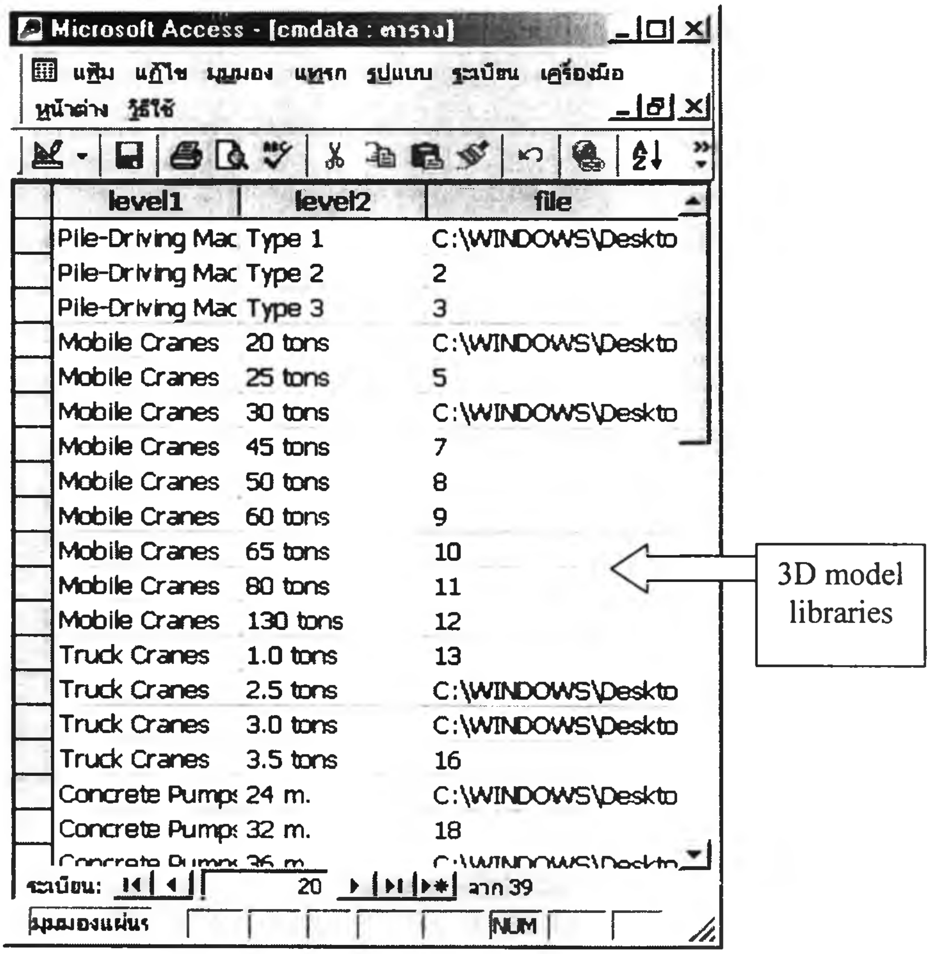
5.5.4.1 Database Structure

The database structure of the Integrated System was designed by separating the system data into several levels. Each level of the system data was linked together and allowed the users to conveniently search and retrieve.

5.5.4.2 Software Implementation

The software used for developing the Integrated System database consisted of three commercial software packages namely 1) Visualizer Scripts in 3DS MAX®, 2) Microsoft Visual Basic®, and 3) Microsoft Access®.

Microsoft Access was applied to generate the data table of the 3D CAD model library. For example, the data table of the 3D CAD model of construction machine and equipment consists of two levels. The first level in the data table is a level of construction machine type, e.g., pile-driving machine, mobile crane, and concrete pump. The second level is a level of the machine model type, e.g., 25-ton model mobile crane, and 24.0-meter model concrete pump as shown in Figure 5.23.



level1	level2	file
Pile-Driving Mac	Type 1	C:\WINDOWS\Deskto
Pile-Driving Mac	Type 2	2
Pile-Driving Mac	Type 3	3
Mobile Cranes	20 tons	C:\WINDOWS\Deskto
Mobile Cranes	25 tons	5
Mobile Cranes	30 tons	C:\WINDOWS\Deskto
Mobile Cranes	45 tons	7
Mobile Cranes	50 tons	8
Mobile Cranes	60 tons	9
Mobile Cranes	65 tons	10
Mobile Cranes	80 tons	11
Mobile Cranes	130 tons	12
Truck Cranes	1.0 tons	13
Truck Cranes	2.5 tons	C:\WINDOWS\Deskto
Truck Cranes	3.0 tons	C:\WINDOWS\Deskto
Truck Cranes	3.5 tons	16
Concrete Pump	24 m.	C:\WINDOWS\Deskto
Concrete Pump	32 m.	18
Concrete Pump	36 m.	C:\WINDOWS\Deskto

Figure 5.23: Data table of construction machines created by Microsoft Access®.

After the structure of the database was generated by the database software, Microsoft Visual Basic was employed to create a window for user interface in order to search data files of 3D models, which were stored in libraries or folders as shown in Figure 5.25. In this window, users can search the data files of the 3D CAD models by pressing a button of data, e.g., the button of construction machines.

In order to retrieve that window, a starting floater was generated by Visualizer Scripts as shown in Figure 5.24. This starting floater became the window for the users to link to the requirement information in the database. Visualizer Scripts was also used to generate the floater of the data sub-tree, which can assist the users to easily search the requirement information as shown in Figures 5.26, 5.27, 5.28 and 5.29.

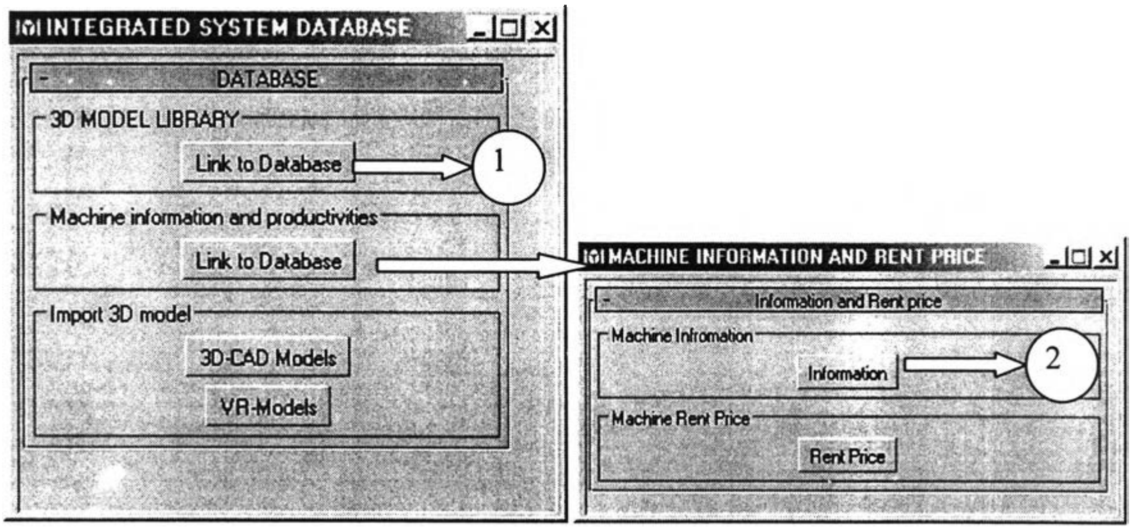
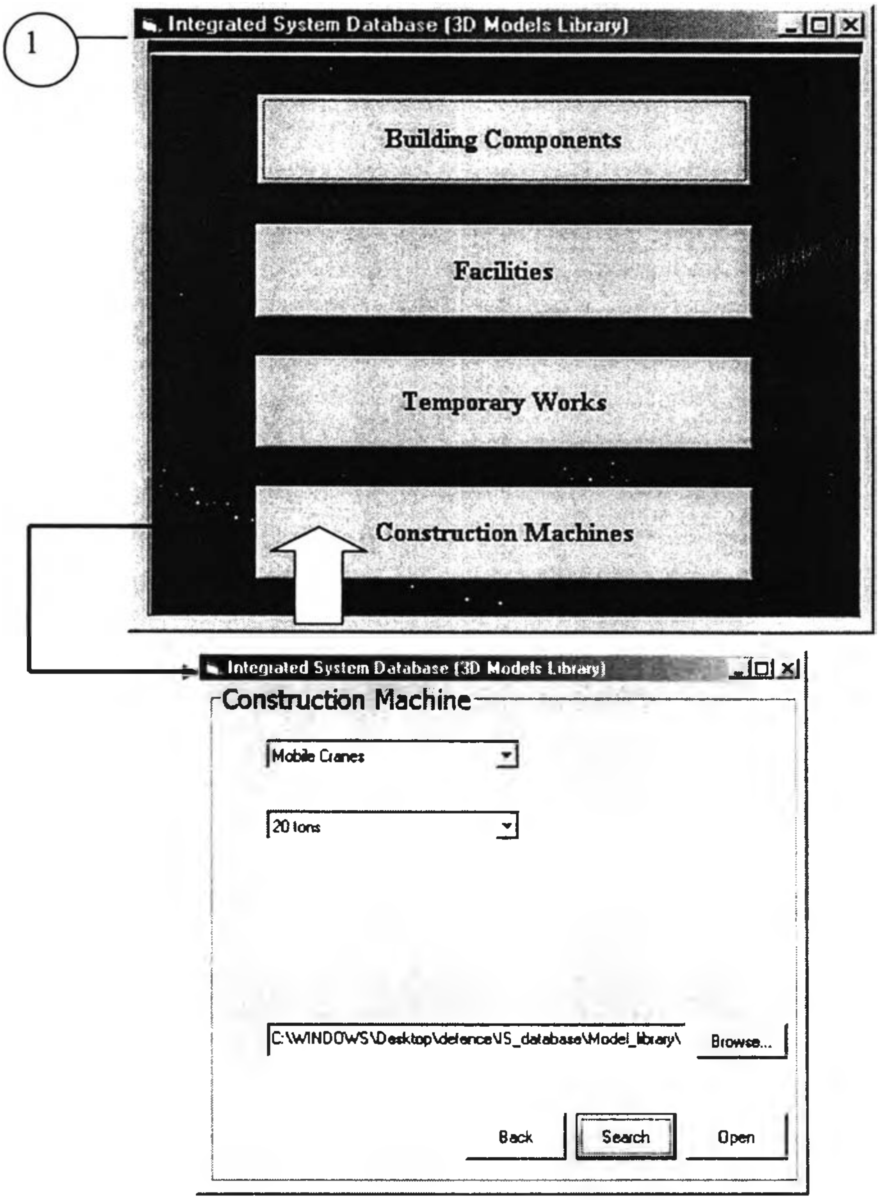


Figure5. 24: The floater of the starting window for linking to the requirement information



Mobile Cranes

20 tons

C:\WINDOWS\Desktop\defence\IS_database\Model_library\

Figure 5.25: Windows of user interface used to search 3D CAD models from library

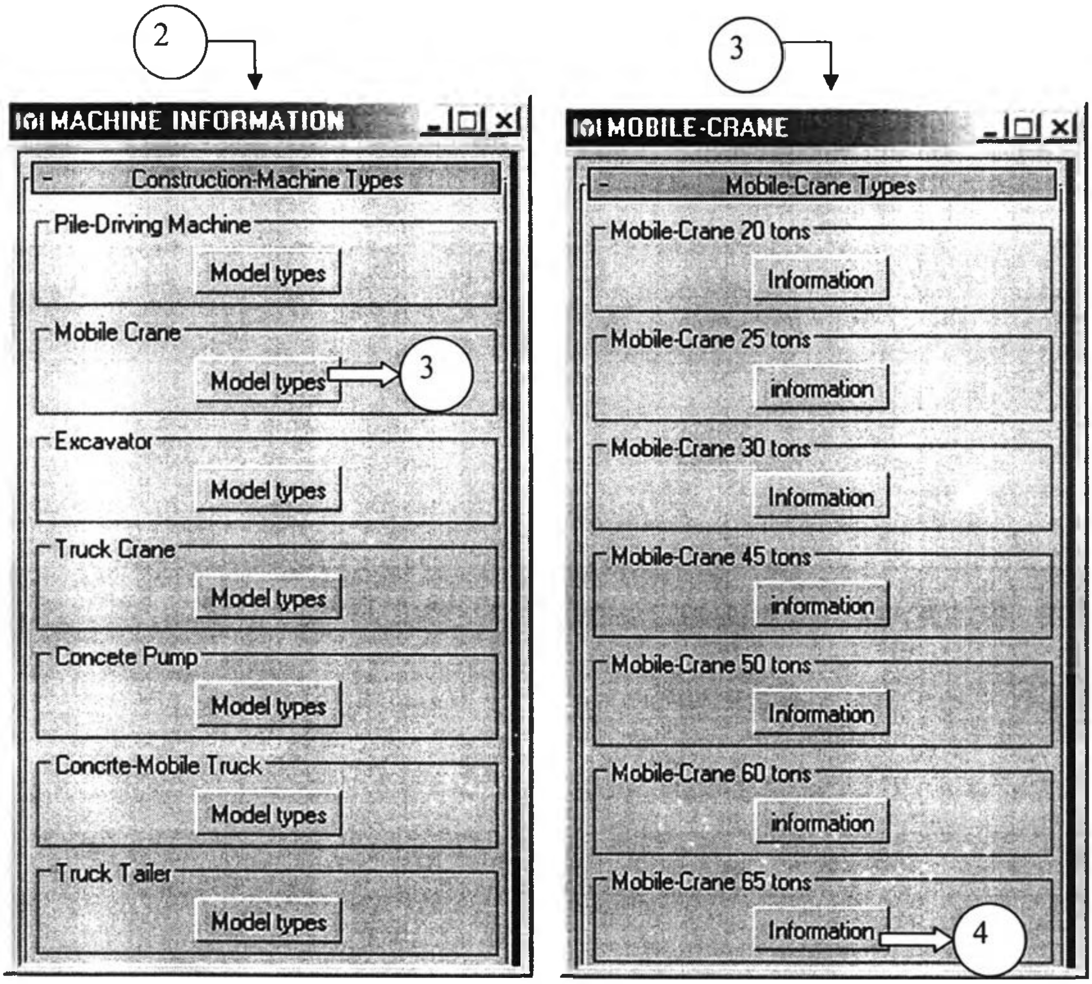


Figure 5.26: Examples of sub-tree float used to link to the requirement information

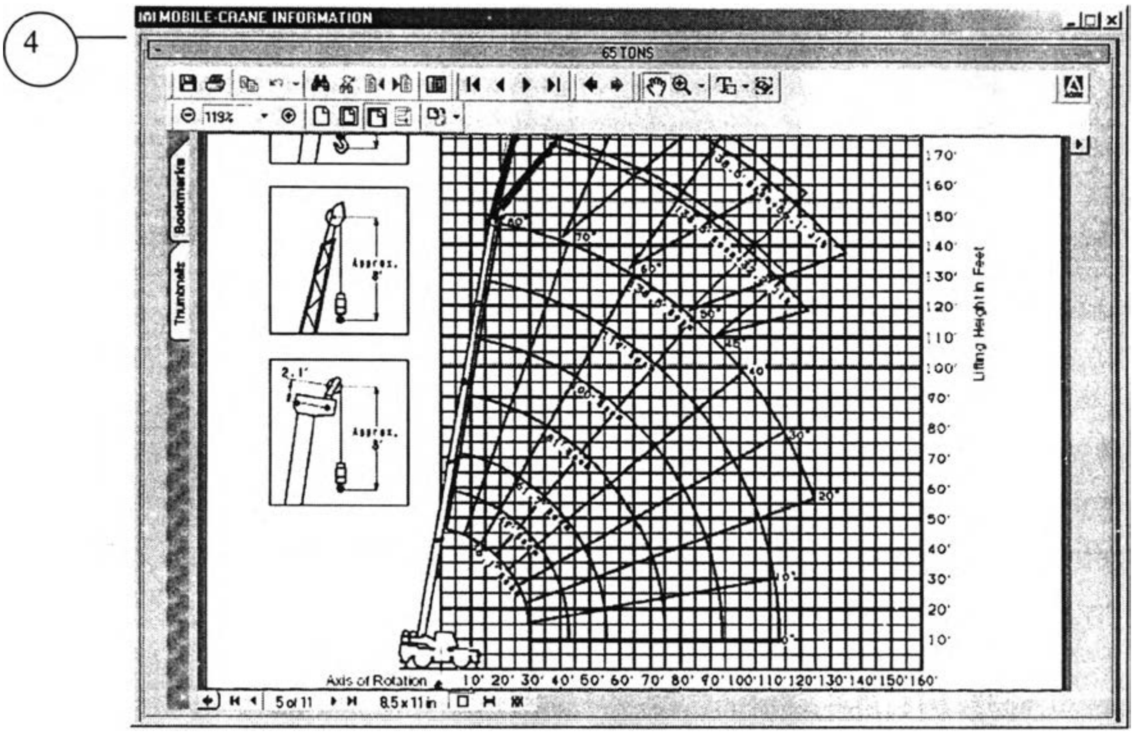
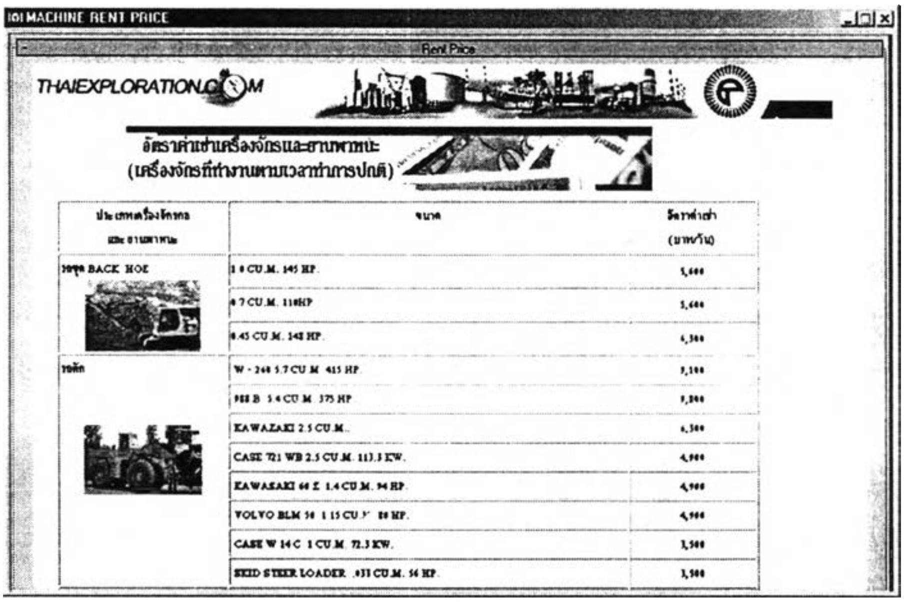


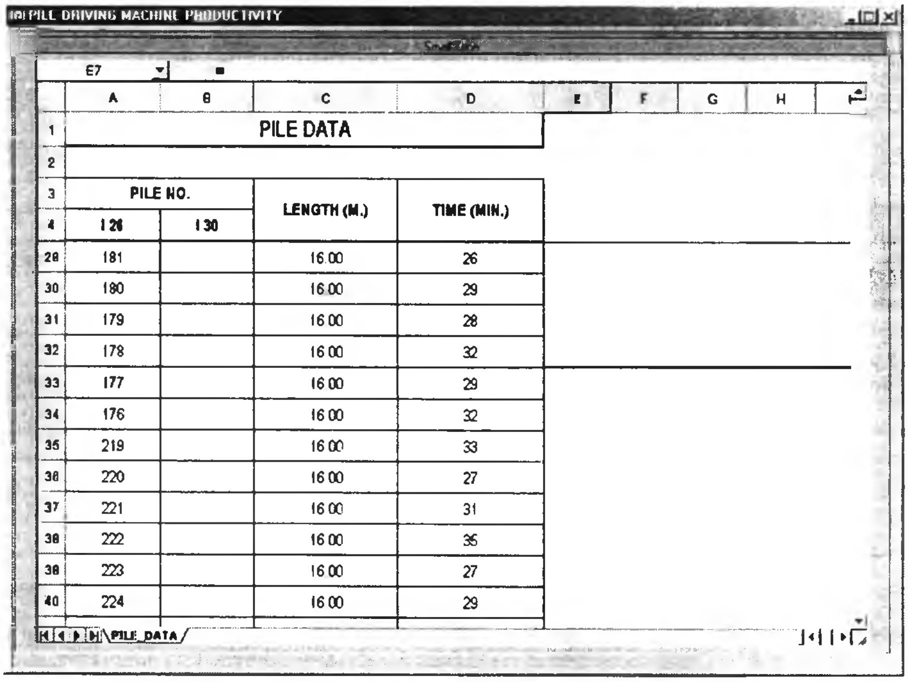
Figure 5.27: An example of requirement information (Mobile crane model 65 tons)



อัตราค่าเช่าเครื่องจักรและยานพาหนะ
(เครื่องจักรที่ทำงานตามเวลาทำการปกติ)

ประเภทเครื่องจักรกล และยานพาหนะ	ขนาด	อัตราค่าเช่า (บาท/วัน)
รถขุด BACK HOE	1.0 CU.M. 145 HP.	5,600
	0.7 CU.M. 110HP	5,600
	0.45 CU.M. 148 HP.	6,300
รถตัก	W - 260 5.7 CU.M 415 HP.	9,100
	988 B 5.4 CU.M 375 HP	9,800
	KAWAZAKI 2.5 CU.M..	6,500
	CASE 721 WB 2.5 CU.M. 113.3 KW.	4,900
	KAWASAKI 60 Z 1.4 CU.M. 94 HP.	4,900
	VOLVO BLM 50 1.15 CU.M. 80 HP.	4,900
	CASE W 14 C 1 CU.M 72.3 KW.	3,500
	SKID STEER LOADER .031 CU.M. 56 HP.	3,500

Figure 5.28: An example of construction-machine rental price



PILE DATA

	PILE NO.		LENGTH (M.)	TIME (MIN.)
	I 26	I 30		
29	181		16.00	26
30	180		16.00	29
31	179		16.00	28
32	178		16.00	32
33	177		16.00	29
34	176		16.00	32
35	219		16.00	33
36	220		16.00	27
37	221		16.00	31
38	222		16.00	35
39	223		16.00	27
40	224		16.00	29

Figure 5.29: An example of information about the productivities of pile-driving machines

5.6 Integrated System Programming

In order to develop the Integrated System, visualizer software called "3D Visualizer" was selected for simulating building construction. One step of system development is to determine the appropriate programming language for operating the virtual software. Then, that programming language must be learnt and applied to develop the system.

This chapter explains the key steps in developing the Integrated System, as follows: 1) defining the system components that will be developed; 2) selecting the appropriate programming language and learning how to apply it; 3) applying that language to program the simulation process of the building construction.

5.6.1 System Components

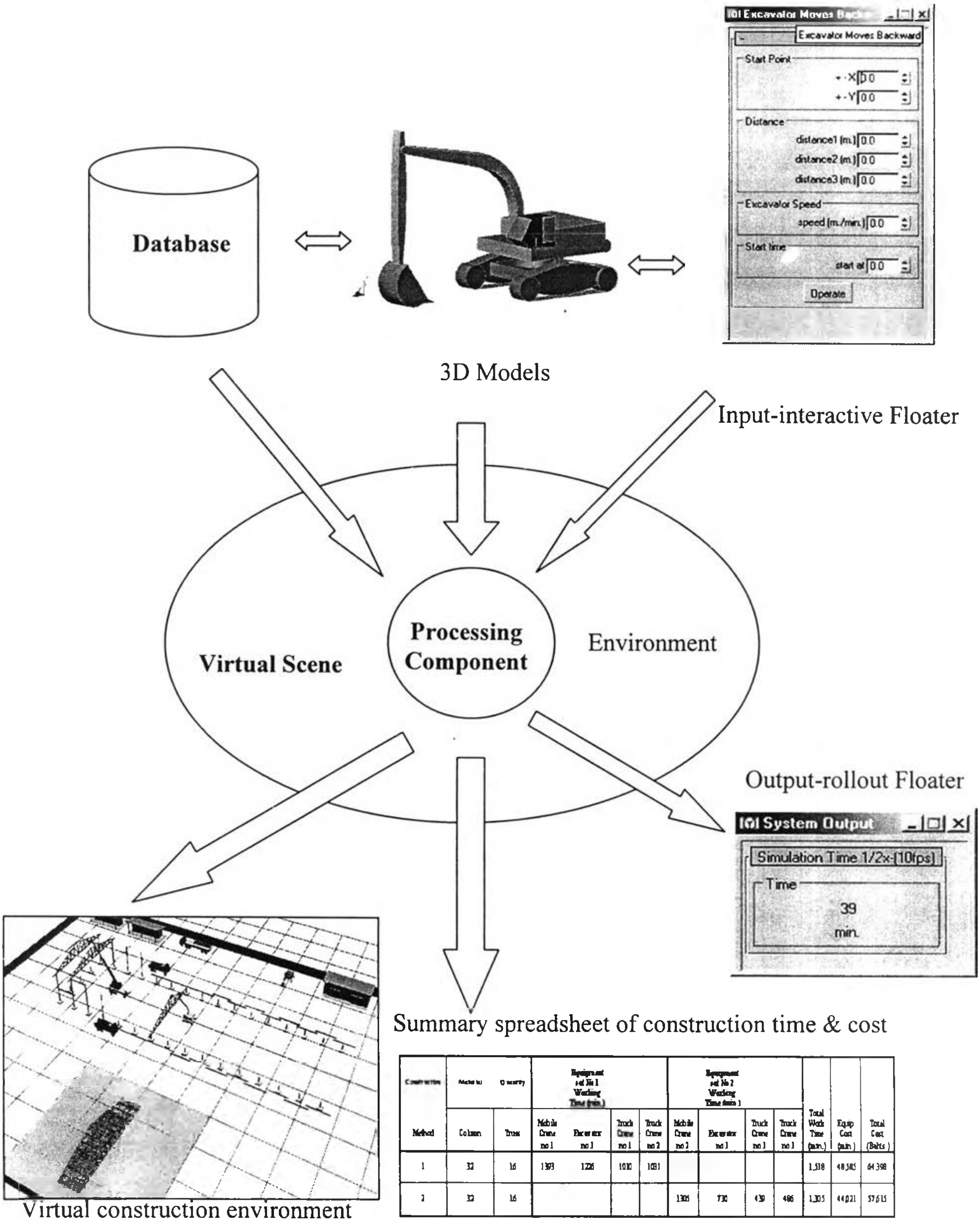
The general system components should consist of three parts: 1) the input component, 2) the system processing component, and 3) output component. In addition the system requirements is the "environment". Similar to the other systems, the Integrated System in this research also consists of three components and an environment. The input component of the Integrated System can be classified into three phases: 1) 3D Dynamic models generated by CAD (described in Chapter3); 2) the construction database; and 3) the interactive floater or direct input window (to be described in this chapter). The processing component is an important part of the Integrated System that consists of system process operations such as variable defining, mathematical factions, controlling system flow (If cause operation, loop operation), and so on. This processing component will be generated and programmed using Visualizer Script language. Since the output components are used to illustrate the results of the system processing, the outputs of the Integrated System should illustrate the building-construction process and methods, time and cost. The building-construction process is illustrated by using virtual scene in 3D Visualizer. The floating windows created by Visualizer scripts are used to illustrate simulation time. Construction costs of different construction methods and resources are summarized using a spreadsheet. The environment of the Integrated System refers to the virtual environment such as the virtual scene or background of the construction site, which helps to illustrate the realistic construction operation. All of these components are present in Figure 5.30.

Visualizer Scripts language is the program language which is built-in and can control the operation of 3D Visualizer. Consequently, Visualizer Scripts was chosen for use in programming and developing the Integrated System components. However, Visualizer Scripts consist of various and complex syntaxes and notations. Thus, only the important syntax should be selected for programming and developing the system. In the following topics, the important Visualizer Scripts language structures are explained by adapting from Visualizer Scripts references.

5.6.2 Visualizer Script Language Structures

5.6.2.1 Visualizer Script Grammar

The Visualizer Script grammar is presented and explained in standard notation called "EBNF". The EBNF (Extended Backus-Naur Form) is used for syntax rules and definitions. These rules typically contain a number of characters with special meanings. For example, brackets enclose optional items, such as the minus sign in front of the number.



Excavator Moves Backward

Start Point
+-X 0.0
+-Y 0.0

Distance
distance1 (m.) 0.0
distance2 (m.) 0.0
distance3 (m.) 0.0

Excavator Speed
speed (m./min.) 0.0

Start time
start at 0.0

Simulation Time 1/2x-(10fps)

Time
39
min.

Construction Method	Material Quantity		Equipment set No 1 Working Time (min.)				Equipment set No 2 Working Time (min.)				Total Work Time (min.)	Equip Cost (min.)	Total Cost (Baht)
	Column	Truss	Mobile Crane no 1	Excavator no 1	Truck Crane no 1	Truck Crane no 2	Mobile Crane no 2	Excavator no 1	Truck Crane no 1	Truck Crane no 1			
1	32	16	1393	1226	1010	1031					1,518	48,585	64,398
2	32	16					1305	730	439	486	1,305	44,021	57,615

Figure 5.30: Integrated System components generated by Visualizer Script language.

Braces enclose items that can be used repeatedly, and bars separate multiple items from which can only be chosen. Sometimes, rules are given names so they can be referred to in the documentation or as parts of other rules. The special characters in the rules are described in Table 10.1.

Table 5.7: Notation of EBNF (Source: MAX Script Reference)

<table>
<tr><th>Notations</th><th>Description</th></tr>
<tr><td>[...]</td><td>items inside the brackets are optional</td></tr>
<tr><td>(...|...|...)</td><td>choose one of the items separated by the bars</td></tr>
<tr><td>{...}</td><td>specify the braced item ZERO or more times</td></tr>
<tr><td>{...}+</td><td>specify the braced item ONE or more times</td></tr>
<tr><td>::=</td><td>define a name for a syntax rule</td></tr>
<tr><td><rule></td><td>you can insert what is defined by the named rule</td></tr>
<tr><td>**bold_characters**</td><td>characters or token as written</td></tr>
</table>

The examples of Visualizer Script grammar are described as follows:

```
<path_name> ::=  $<path> | $
```

This grammar "`$`" is used as a symbol in front of the path names of objects in 3D Visualizer, for examples, **`$BOX01`**, **`$Truck_body02`** or **`$wh01`**.

```
<number> ::=  [-]{<digit>}[.{<digit>}[(e | E)[+ | -]{<digit>}+]
```

This grammar is used for defining the decimal numbers, for example 10.00, -5.18, 10.0e2, 1.06E6 or -12.05E-4.

5.6.2.2 3D Visualizer Command

The Visualizer Scripts can operate 3D Visualizer by using several commands. The general syntax for 3ds max commands is:

```
max <command_name>
```

The example is:

```
max file import
```

This command used to activate a file, import and display selected file to import dialog of 3D Visualizer.

5.6.2.3 Name, Literal Constants and Expression

Name

Names are used in Visualizer Script to identify variables, functions, parameters, properties, and so on. Names start with an alphabetic character or "_" (underscore), and can contain any number of alphanumeric characters or "_". The examples of name in Visualizer Script are `name01`, `Excavator_body` or `Truck_wheel002`.

Literal constants

Visualizer Script has many data types built-in. The syntax of the literal constant specifies both its value and data type. The literal constants are divided into number literal, string literal, time literal, pathname literal, 2D and 3D points literal and array literal. The examples of time literal are `1m15s`, which means 1 minute 15 seconds and `2m30s5f2t` means 2minutes 30 seconds 5 frames 2 ticks. Examples of 3D points literal are `[2,5,8]`, `[2.3,4.5,6.8]` or `[sin30, cos15, sin60]`.

Expression

Visualizer Script is an *expression-based* language. Every construct in the language is an expression and yields a result; this includes constructs that other languages consider statements. The simplified syntax of the language makes it possible to build very expressive code. The most important expressions for generating the Integrated System are 1) math expressions, 2) comparison expressions, logical expressions and block expressions. Examples of the notation of math expression are:

```
<math_operand> + <math_operand> -- standard arithmetic addition

<math_operand> - <math_operand> -- standard arithmetic subtraction

<math_operand> * <math_operand> -- standard arithmetic multiplication
```

Examples of the notation of comparison expression are:

```
<compare_operand> == <compare_operand>   -- equal

<compare_operand> != <compare_operand>   -- not equal

<compare_operand> > <compare_operand>    -- greater than
```

Examples of the notation of logical expressions are:

```
<logical_operand> or <logical_operand>  -- true if either operand is true

<logical_operand> and <logical_operand> -- true if both operands are true
```

5.6.2.4 Controlling Program Flow

Visualizer Script has several <expr> expression forms used to explicitly control the flow of execution. These expression forms correspond to the standard control constructs found in many programming languages. The scripts used to control the program flow are 1) if expression; 2) case expression; 3) while and do loops; 4) for loop; 5) try expression; 6) skipping loop and 7) exit loop.

Examples of syntax for controlling program flow are:

```
if <expr> then <expr> [else <expr> ]

while <expr> do <expr>

for <var_name> ( in | = ) <sequence> ( do | collect ) <expr>
```

5.6.2.5 Time and Key Function

The 3D object or a collection of 3ds max objects can be controlled by the controller time and key functions. Some of the time and controller functions that work this way are:

`DeleteTime, reverseTime, scaleTime, insertTime, setTimeRange, addNewKey, deleteKeys, selectKeys, deselectKeys, moveKeys, sortKeys, reduceKeys.`

An example of time and key function is:

```
key = addNewKey ball.pos.Zposition.controller 0

key.outTangentType = #slow

key.value = 50
```

5.6.2.6 Animation Controller

In order to animate the motion of 3D objects using Visualizer Scripts language, several types of animation controllers are used for controlling these virtual objects follow their real actions. However, only a few animation controllers are used to animate the 3D objects of building construction activities in this research, namely 1) Position controller; 2) Rotation controller; 3) Scale controller; 4) Link controller; 5) Bezier controllers; 6) Linear controller and 7) TCB controller. The 1st – 4th controllers are the main controllers used to control the 3D models and visualize their motions. But the 5th – 7th controller are specific controllers that can be used to assign the patterns of 3D model's movements, for examples Bezier_position, Linear_rotation or TCB_scale.

The examples of Script for controlling the 3D object (named truck) are presented as follows:

```
c = bezier_position ()              -- create and assign new controller
$truck01.pos.controller = c
k = addNewKey c 0f                  -- add a key at frame 0
k.value = [10,0,0]                  -- set value there
k.outTangentType = #slow            -- and outgoing tangent type
```

5.6.2.7 Creating Utilities and Rollouts

Visualizer Script provides a set of classes and functions and some special syntax to allow construction of rollouts that can be incorporated into the existing 3D Visualizer user interface. A scripted utility panel is created using the utility definition constructed in Visualizer Script. The top-level syntax is as follows:

utility <var_name> <description_string>

The rollout floater windows can be created with the **newRolloutFloater()** function. These windows appear on the desktop and function in a way similar to the Material Editor or the Selection floater in 3D Visualizer. The individual rollout can be created for these windows using the rollout definition constructed in Visualizer Script. They can be installed in a rollout floater window using the **addRollout()** function. Examples of scripts for creating the utility and rollouts are as follows:

```
rof=newrolloutfloater "test" 200 200
addrollout ui_item rof
rof.size=[300,300]
```

5.6.3 Programming Methodology by Visualizer Scripts

After the 3D models were imported to 3D Visualizer and assigned their mechanisms, those models were controlled by Visualizer Scripts programming. The components of 3D models are controlled to perform realistic action by changing their actions following simulation time advance.

For programming construction activities by Visualizer Scripts, the key components of programming need the following assignments:

1) The components of 3D models must be assigned their name according to *name syntaxes*, for examples "`$wh10`" is assigned as the name of the excavator's track-wheel, "`$bucket01`" is assigned as the name of the excavator's bucket, "`$truss01`" is the name of roof truss no.01, and so on.

2) Mathematical calculations need variable assignments, for example, “`n`” is equal to numbers of cycle time, “`start`” refers to start time, and so on.

3) Assigning mathematical function or equation, for example:

```
N = tw/vol
```

N = Amounts of operation cycles
tw = total work done (m^3)
vol = volumn of bucket (m^3)

The programming needs the start time values for animating 3D models. Start time values are put through the programming by using input windows (rollout floater) that will be explained in a later topic. This start time value is compared with the simulation clock value that is running. If the start time value is equal to the simulation clock value, the simulation and animation process will start. If not, the process will be waited. The simulation process starts by generating a random number. That random number is used for reading the productivity rates from database. These productivity rates are used to calculate the cycle time of the construction action. This cycle time is broken-down into minor operation times according to the actual operation process. For example, the cycle time of earth excavation by an excavator is divided into minor operation times as follows:

$t_{Excav} = \mathrm{cyct}_{01} + \mathrm{cyct}_{02} + \mathrm{cyct}_{03} + \mathrm{cyct}_{04} + \mathrm{cyct}_{05} + \mathrm{cyct}_{06} + \mathrm{cyc}_{07} + \mathrm{cyc}_{08} + \mathrm{cyc}_{09}$
cyct_{Exc} = cycle time of earth excavation by excavator
cyct_{01} = time of forward rotation of excavator’s body
cyct_{02} = time of forward rotation of excavator’s 1st arm
cyct_{03} = time of forward rotation of excavator’s 2nd arm
cyct_{04} = time of forward rotation of excavator’s bucket
cyct_{05} = time of backward rotation of excavator’s 1st arm
cyct_{06} = time of backward rotation of excavator’s body
cyct_{07} = time of backward rotation of excavator’s 2nd arm
cyct_{08} = time of backward rotation of excavator’s bucket
cyct_{09} = time of forward moving of excavator

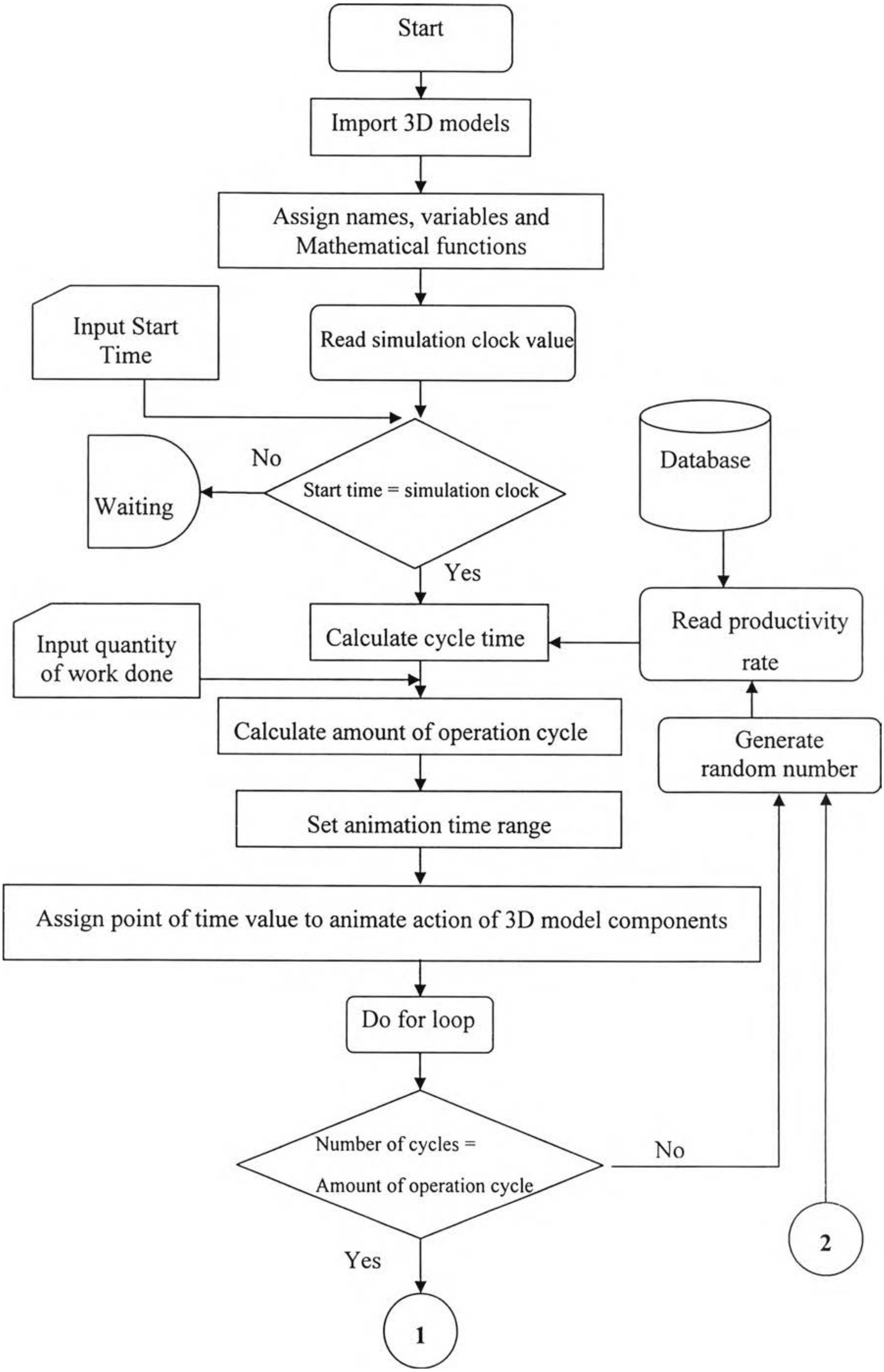


Figure 5.31: Flowchart for programming construction activities

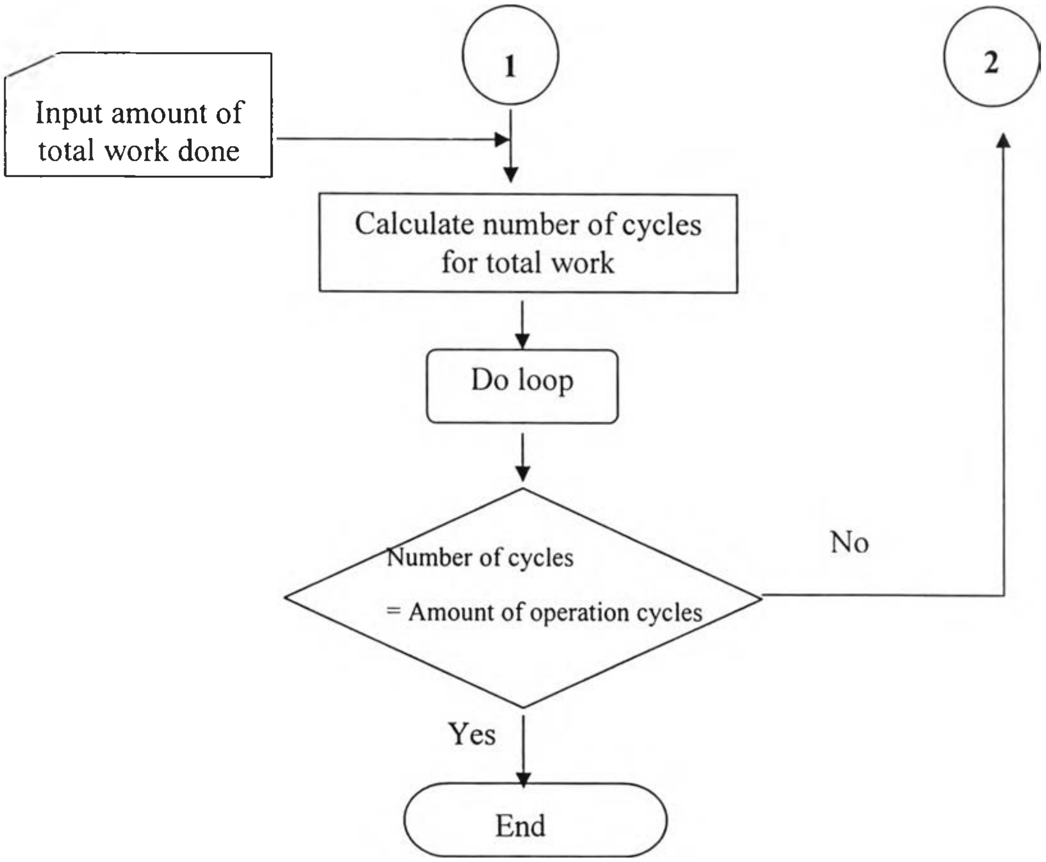


Figure 5.32: Flowchart for programming construction activities *(Continued)*

The quantity of work done is put through the programming to calculate amount of operation cycles. After that, the animation process starts by setting the animation time range and assigning point of time values to animate the actions of the 3D model components. If the numbers of cycle operation are greater than one, a loop operation is operated in the beginning step until the numbers of operation cycles are equal to the amount of operation cycles. For animating all of the construction work, the amount of total work done is put through the programming. The simulation processes are operated until they have covered all of the construction work. All of the simulation and animation processes that were explained above are illustrated by the flowchart in Figures 5.31 and 5.32.

5.6.4 Controlling Construction Activity by Visualizer Scripts

5.6.4.1 Time Measurement

The Visualizer Scripts language can measure time in floating points of time called "animation time units". Animation time can be run depend on frame rate in 3D Visualizer. One unit of animation time can equal the construction duration that is most suitable for animation, e.g., minutes, hours, days, weeks or months. These durations can be set to be equal to the frame rate in 3D Visualizer. The following examples are ratios of animation times and frame rates.

1 frame = 1 minute
10 frames = 1 minute
100 frames = 60 minutes or 1 hour
100 frames = 1 day

When 3D Visualizer is used to animate construction duration, the frame rate can be set to increase or decrease the speed of animation. The speed of the construction process will increase two times when the speed of animation is set to be 2x, and the speed of construction process will increase four times when the speed of animation is set to be 4x (Normal speed is 1x). Conversely, the speed of the construction process will decrease two times when the speed of animation is set to be 1/2x, and the speed of the construction process will decrease four times when the speed of animation is set to be 1/4x.

5.6.4.2 Controlling Construction Activities by Visualizer Scripts

In order to control construction process flow using Visualizer Scripts, according to the flowcharts in Figures 5.31 and 5.32, the cycle time of construction activities is defined and divided into several minor time ranges. All of the time ranges are used to define animation time points, and these time point values are used to set the action time of 3D models in construction activity. In the above example, the cycle time of an excavator for earth excavation is divided into nine minor time ranges (cyct$_{01}$ to cyct$_{09}$). Those time ranges are used to define the animation time point values and set action of the excavator model's components, as presented by the following Script variables and equations.

```
-- t          = cycle time of earth excavation by excavator
-- cyct01     = time of forward rotation of excavator's body
-- cyct02     = time of forward rotation of excavator's 1st arm
-- cyct03     = time of forward rotation of excavator's 2nd arm
-- cyct04     = time of forward rotation of excavator's bucket
-- cyct05     = time of backward rotation of excavator's 1st arm
-- cyct06     = time of backward rotation of excavator's body
-- cyct07     = time of backward rotation of excavator's 2nd  arm
-- cyct08     = time of backward rotation of excavator's bucket
-- cyct09     = time of forward moving of excavator
st = 0 --start time value
```

```
t = cyct01+cyct02+cyct03+cyct04+cyct05+cyct06+cyct07+cyct08+cyct09
-- 1st to 9th time point values are defined belows:
n = st
m = n+cyct01
o = m+cyct02
p = o+cyct03
q = p+cyct04
r = q+cyct05
s = r+cyct06
u = s+cyct07
v = u+cyct08
w = v+cyct09
```

The Visualizer Scripts command used to define the total time range for animating a 3D models movement is "*AnimationRange*". The syntax of the *AnimationRang* command is:

```
AnimationRange = Interval t1 t2
```

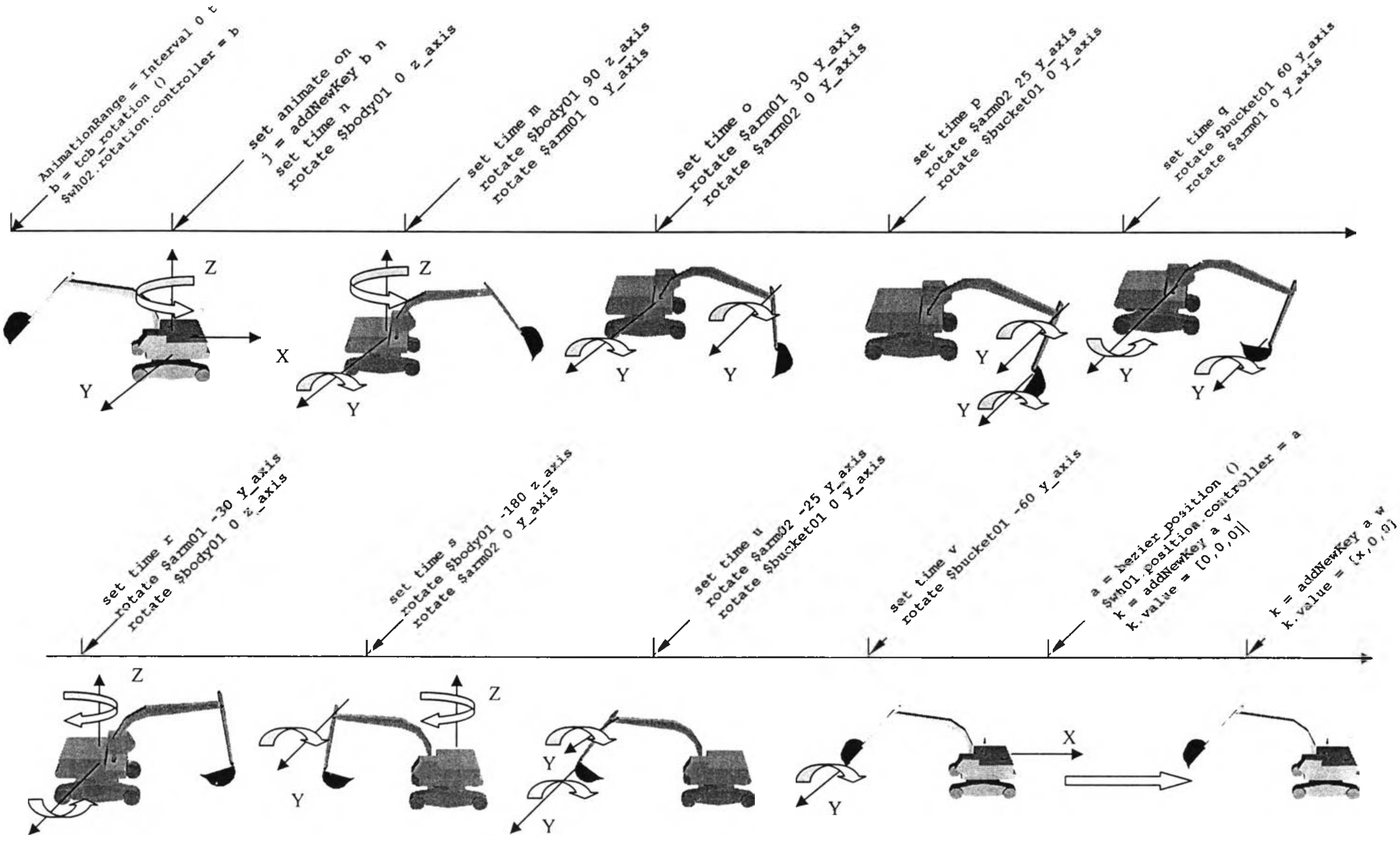
In order to animate earth excavating by the excavator, each component of the excavator model are controlled by the rotation controller and position controllers. The syntax for assigning the type of the controller are:

```
a = bezier_position ()
$wh01.position.controller = a
b = tcb_rotation ()
$body01.rotation.controller = b
```

The Bezier position is one type of position controller that was used to control the excavator position and the TBC rotation is one type of rotation controller that was used to control the body of the excavator model named "body01".

The syntax used to assign each time point value to each component of the model are:

```
set animation on
j = addNewKey b n
set time n
rotate $body01 0 z_axis
set time m
rotate $body01 90 z_axis
```



```
AnimationRange = Interval 0 t
b = tcb_rotation ()
$wh02.rotation.controller = b

set animate on
j = addNewKey b n
set time n
rotate $body01 0 z_axis

set time m
rotate $body01 90 z_axis
rotate $arm01 0 Y_axis

set time o
rotate $arm01 30 Y_axis
rotate $arm02 0 Y_axis

set time p
rotate $arm02 25 Y_axis
rotate $bucket01 0 Y_axis

set time q
rotate $bucket01 60 Y_axis
rotate $arm01 0 Y_axis

set time r
rotate $arm01 -30 Y_axis
rotate $body01 0 z_axis

set time s
rotate $body01 -180 z_axis
rotate $arm02 0 Y_axis

set time u
rotate $arm02 -25 Y_axis
rotate $bucket01 0 Y_axis

set time v
rotate $bucket01 -60 Y_axis

a = bezier_position ()
$wh01.position.controller = a
k = addNewKey a v
k.value = [0,0,0]

k = addNewKey a w
k.value = [x,0,0]
```

Figure 5.33: Syntaxes of Visualizer Scripts command used to control an excavator movement for earth excavation

The time command waits for the simulation clock to reach the time value that was assigned to the component of the model. The Visualizer scripts then execute the commands that follow it until other the command is reached again. In the above Visualizer scripts commands, new key-frames are generated by "*addNewKey*" as soon as the simulation clock is equal to *n* value. At this time value, the body of the excavator is assigned a rotation angle is equal to zero degrees around the Z-axis. When the simulation clock changes to *m* value, the body of the excavator is assigned a rotation angle equal to 90 degree around the Z-axis instead. Assuming that 1 frame is equal to 1 minute of animation time. The body of the excavator spends (*m-n*) minutes to rotate 90 (90-0) degrees around the Z-axis. The other key frames are generated at the other time points until the total cycle time of this excavation has been covered as shown in the Figure 5.33.

5.7 Conclusions

This chapter presents the key steps and methods of Integrated System development. The important methods for creating the visualizing construction activities are described. Time and event relationship is explained. Then, in order to make sure the actions of the main components (materials and equipment) of construction activities are correct, the visualizing controllers called "controlling parameters" are assigned to these components. The controlling parameters consist of: 1) the transformation parameter (position, rotation and shape-transformation), 2) the visibility parameter, and 3) the linking parameter. In order to illustrate the visualizing construction activities according to the simulation time, the frame update technique is explained in two categories: 1) key-frame, and 2) frame rate. Finally, pile driving is presented as an example of the factory construction activity.

Basic simulation processing and methods can be applied for developing the Integrated System. Discrete-event simulation is an appropriate simulation method for simulating construction operation. The time advance mechanism in the simulation system was explained, and a time advance approach called "next-time event time advance" was selected as an appropriate approach for the simulation of construction operations. The "Monte Carlo technique" is a popular simulation technique used in solving several problems. The random numbers and accumulated probabilities of the requirement data are important elements used in the process of the "Monte Carlo simulation technique". The random numbers generated by computer programming should be validated for uniformity and independence. Five statistical methods are used to validate the uniformity and independence. The building construction data that related to construction time and cost are classified and converted to accumulate probability that can be used within the Monte Carlo simulation technique.

In addition, this chapter explains the main steps and methods used to design, develop, and implement the system database, which is used as an input of the Integrated System and applied to simulate the virtual construction. The methods for developing the Integrated System database in this research are divided into four processes as follows: data classification, data identification, data acquisition and collection, and database development and implementation.

This chapter also explains the important steps for programming the Integrated System as follows: 1) defining the system components that will be developed, 2) selecting the appropriate programming language and learning how to apply it, and 3) applying that language to program process simulation of the building construction. The Integrated System consists of three important components: 1) input component, 2) processing component, and 3) output component. These components were programmed by using Visualizer Scripts language. The structure of Visualizer Scripts language and methods of programming are described. Finally, the methods for controlling the construction activities by using Visualizer Scripts commands and syntaxes are explained by using an example of earth excavating by an excavator.