A Management Game for Evaluating the Selection of Project Plans in Construction

S.H. Al-Jibouri
University of Twente, Enschede, The Netherlands
s.h.al-jibouri@sms.utwente.nl

M.J. Mawdesley
School of Civil Engineering, University of Nottingham, UK
Michael.Mawdesley@nottingham.ac.uk

ABSTRACT: Project planning is an essential task of project management that is taught in most undergraduate and postgraduate programmes within universities. However unlike in engineering or other technological areas, students in the management of engineering projects will probably not have the opportunity during their study to test and employ the concepts they have learned about this subject.
Project planning is practical subject and previous research illustrates that traditional construction management instruction methods and techniques are insufficient to equip students with skills to solve problems in the real world. There is a belief that management games represent a viable alternative to solving this problem.
This paper describes an experimental management game that allows players to choose their own planning activities to complete a construction project. The game then makes a judgement as to how good the set of activities are.
The work develops theories using tree-like structures to represent project plans. This theory enables a comparison of the players' plan and a standard plan and a consequent judgement of its goodness. The theory is tested in an experimental computer game. The development and testing of this game are presented and discussed. Conclusions about the feasibility of such a game are presented.

KEYWORDS: Construction, Management Game, Planning, Simulation

1. INTRODUCTION

More and more Universities are offering courses in the area of construction engineering and management (AbouRizk, 1994). Traditionally, the instruction methods used in these student' curricula rely on exposing the student to applied science courses relevant to the construction industry. These courses and their application to certain deterministic decision models form the basis of the construction engineers training (Sawhney, 1998, Halpin, 1976).

This alone, however, is insufficient to equip students with skill to solve problems encountered in the real world of construction or to convey complex engineering knowledge effectively, see (AbouRizk, 1994). Also curricula often convey knowledge in fragments in a series of courses (Fruchter, 1996).

This means there is a gap in knowledge and experience between the 'applied science' graduate engineer and the 'real world' construction engineer. Traditional arguments state that experience is the only appropriate basis for filling this gap (Tatum 1987). Therefore it is essential that students get 'on-the job' training. This training however is expensive and as (Cullingford et al, 1979) pointed out "it is often impractical to give students access to full-scale projects, where the cost and timescale clearly prelude experimentation". This leaves the graduate and the employer in a mutually frustrating relationship. Management cannot give important job site problems to the inexperienced graduate to solve due to the high cost of errors involved should he or she be wrong. So important decisions are restricted until years of experience have been gained, consequently the employer bears the cost of the engineers extra education (Halpin, 1976). Furthermore the time for the graduate to fulfil his or her potential is drawn out. This isn't a positive position for employer or graduate.

In the 1960's Au and Parti proposed that construction games might be a way of improving traditional methods of teaching, by using " …Computerised heuristic games portraying social, economic, and technological decisions…for the education of engineers and planners who are engaged in works directly or indirectly related to the construction industry" (Au, 1969).

2. THE DEVELOPED GAME

2.1 Game objectives

The work described here represents a management game that has been developed in order to teach students at the University to use their knowledge and experience to plan activities for a project. The players use the game to test their experiences in the planning of a specific construction project.
In terms of the construction management course, the game is intended to have the following teaching aims:

- To teach activity allocation : what activities are needed to complete the project.
- To teach resource allocation: what resources are needed for a project.
- To teach activity scheduling: activities placed into a network schedule.

The game is still under development and at this stage of the work , only the first aim has been realised.

The game aims on the other hand are :

- To design a game which enables achieving the teaching aims.
- To make the game as realistic as possible.
- To make the game interesting and fun.
- To involve an element of risk and luck.
- To make the game graphic.

2.2 Game description

The game is designed to teach students to use their knowledge and experience to plan activities for a project. The players, given the project plans, must consider what activities are needed and to what detail these activities need to be planned to. The game is a 'model type game', see Elgood (1989), in the sense that there are standard project models that the game is based on. However it is different in that it doesn't give the players options to choose from but the player must realise their options.
The game is written in Pascal and was developed in the Borland Delphi IDE.

2.3 Game infrastructure

The game, which is tentatively christened as Genesis consists of two parts 'Editor' and 'Player'.

' Editor' allows the player to input his or her own project. The game is designed to allow multiple projects to be used as the games model. This gives the course convenor, or the students, options over the type and difficulty of projects they might want to practice on. A first year student is not going to be at the same level of experience as a postgraduate; therefore it is important that different types of projects are available.

' Editor' also takes the instructor through the steps required to input a project. However the instructor must understand the game before inputting his or her own project because the game requires information about the project structure that can sometimes be difficult to determine.

'Player' is the main part of the program. It consists of three major steps as shown in Figure 1. In the first step, the player is introduced to the game; a PowerPoint show gives an overview of the game. It explains the game, its goals, how it is played and what it checks for as well as how it works. If there are multiple projects to choose from, the player can read a project description and load it up. Once loaded, detailed plans of the construction project are displayed in the second step. The player can study these plans, start to plan the activities and input them into the computer. The player can reference all the plans, a project dictionary, a list of their activities at any time, along with the ability to erase, or edit any activities already inputted, save or open activity plans. Stage 2 is completed when the player submits all the activities for evaluation.

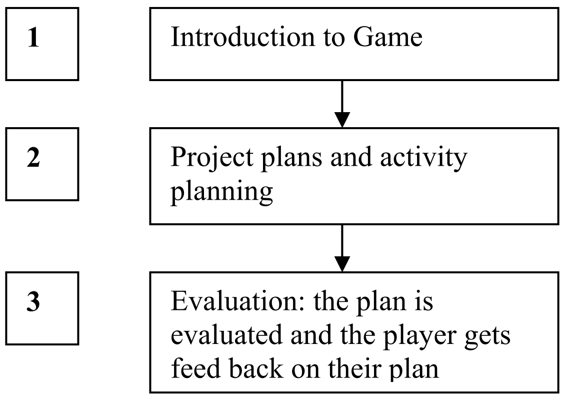
1
Introduction to Game
2
Project plans and activity planning
3
Evaluation: the plan is evaluated and the player gets feed back on their plan

Figure 1. The three stages of the ' Player' section of the game

In stage 3, the computer checks and evaluates the activities and plans chosen and the player is given an overall score relative to these checks. The computer also allows the player to view its evaluation displaying; missing components, false activities, duplicated activities and detail advice.

3. METHODOLOGY

The methodology developed for the evaluation of the plan of work activities is based on theories developed by J. Booth (Booth, 1993). The game has applied some of these theories, developed and adapted them to form a working construction planning game.

The main theory behind this game is the generation of a tree-like structure representing the players plan. This project tree is evaluated using a comparison process against a standard tree. The methodology consists of four stages:

- Standard tree
- Parsing
- Mapping project tree
- Evaluation

These are explained in the following sections.

3.1 Standard tree

Project planning consists of identifying and organising activities. Each activity is a plan of work for a section (or sections) of a project. There are two types of components in an activity.

- Identifiers
- Work Types

An activity is parsed (see section parsing) so that the sentence describing the activity contains only these two types of components. In an activity there is usually only one work type. Multiple work types would suggest multiple activities in one sentence. However there can be multiple identifiers; a typical set of classification categories are:

- Structural Elements (e.g. rafter)
- Structure (e.g. House)
- Section (Area) of structure (e.g. Roof)
- Section (area) identifier (e.g. centre)

Using these components and a set of classifications a tree structure can be created. The components are classified as nodes and the connections between the nodes are classed as branches. In order to structure the tree these classification categories must be prioritised. The priority order depends on what type of orientation the tree should be.

There are many ways of structuring a project tree. There can be a variation in the types of classifications, number of classifications components and the priority ranking of the classification. A tree can be a 'structured oriented' or 'a work oriented'.

To evaluate a players' plan, the game must compare it to a 'standard'. A 'standard project tree' is used to represent this standard plan. The standard tree represents the limitations of the game; although the player is allowed freedom of choice in choosing the activities there must be a set standard so that reasonable evaluation can take place.

Choosing the orientation, classifications, and prioritisation of a standard tree can be difficult. Different tree orientations may be needed to run different games, no one type of standard tree will be suitable for all the project games. In later developments of this game it is hoped that the ' Editor' may be able to choose and select the type of standard tree needed. In this experimental program the editor is restricted to project games that can use the tree structure that the 'Summer-House Game', an example project, is based on.

The ' Summer-House Example Project' standard tree is a structure-oriented tree with the following classifications and in order of priority:

1. Structure
2. Area of Structure
3. Structural Element
4. Work Type
5. Area Identifier

The structure of the standard tree looks like a tree's roots, always branching down from stem. This structure is used to demonstrate a detail hierarchy. All the nodes at each level are represented by a 'rank' corresponding to its classification priority. This rank gives an indication to the project detail included in an activity. The lowest ranked node in an activity represents all the activity nodes branching down below it.

3.2 Parsing

The game fundamental aim is to give the player the choice over activities he or she plans. In order for this to work the computer must be able to understand the activities entered. The method used for this is a well-known technique called parsing.

Parsing resolves a sentence into its components parts and identifies them.

The game parsing program is used twice; once to identify the sentence and the then to map the component parts onto the standard tree. The process consists of three stages.

The first stage relates to the process of separating the activity into words components. For example the word components for activity ‘ Excavate the deck footings’ are:

Excavate	The	Deck	Footings

The word components must be recognised by the game’s standard dictionary.

The English language presents parsing with several problems: words have more than one meaning; different words can have the same meaning; people often abbreviate words; sometimes two different words can have the same abbreviation. This is coupled with each player having his or her own way of describing areas of the project. To deal with these problems the game has to set some limitations on the words players can use. Every project has its own dictionary of words the ‘Editor’ creates. Nevertheless not just one set of standard activity component names, equivalent names can also be included , e.g. ‘dig’ and ‘excavate’ can be included to mean the same work type. However the dictionary does set a limit on the words the player can use.

There are two types of word included in the dictionary: zero-words and node-words. Zero-words can be defined as having no meaning in the standard tree, but are English word used to construct sentences. Words like: ‘the’, ‘to’, ‘a’, etc. [Note; ‘and’ is not included because ‘and’ suggests the start of a new activity.]. these ‘ zero-words’, once identified are dropped from the program. They are not referenced again. The node-words are words that have meaning in the standard tree; ‘dig’, ‘excavate’, ‘erect’, ‘construct’, ‘wall’, ‘walls’, etc

Once the sentence is separated in stage 1, the word components are compared to the dictionary. If a word isn’t recognised by the program as being part of the dictionary, the activity is rejected, the word highlighted, and the player must either reword the sentence using words from the dictionary or exclude that activity.

The third stage of the parsing process relates to assigning standard tree characteristics.

Each word in the dictionary has two identifiers associated with it; Standard word and standard word rank.

Every word has a standard word equivalent, even if it is the same word, and also a rank of the level at which the standard word is in the tree. The parsing stage assigns these identifiers to help the program identify the sentence components in the tree. Zero-words have a rank of zero and are dropped from the program after the third stage.

3.3 Mapping the project tree

The game uses the standard tree as a template and maps the project activities onto it. The project tree is evaluated once all activities are mapped on the standard.

The method starts once all the activities have been inputted. The activities are parsed to get the word components and their identifiers. The words are ordered from the highest rank (1 being the highest) to the lowest.

Every activity, in the plan, has an activity number associated with it. Each node in the tree has an activity array. This array contains all activity numbers of which that node is a component. The lowest node in the activity branch is assigned the activity number to its activity array. Once the lowest standard tree node gets the activity number it maps the number down to all, if any, of the nodes below. The other nodes in the branch are not assigned the activity number but they are then tagged as used (this is represented by a node ‘used’ component that is changed from ‘0’ to ‘1’). Only the lowest node gets the activity number because the other components in the activity are working as identifiers, and may be used in other activities. Although a node activity array can contain more than one activity as shown in Figure 2, ideally they should only contain one. This Figure shows a section, of an example standard tree, where Activity 1 and 2 has been mapped on.

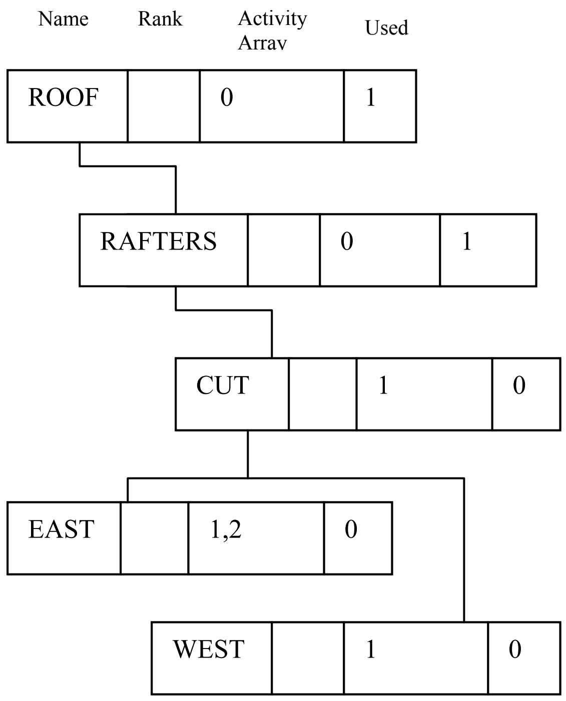

Figure 2. An example of the mapping process within the game.

This method, of mapping the activity down from the lowest node, also allows different levels of management. Sometimes the lowest node in an activity could be the highest node in the tree, e.g. ' Build the Summer-House', therefore only that activity is needed. Once all activities have been mapped onto the standard tree, the tree is evaluated.

4. EVALUATION OF PLANS

The player's plan is evaluated based on the following four areas.

Completeness: This is to check that the player has planned the whole project. Once the activities have been mapped onto the standard tree each node is evaluated. Each node's 'activity array' should contain at least one activity number representing that that component is part of an activity. If there is no activity numbers in the array, the nodes 'used' array is checked to see if the node has been included in an activity. The reason for this 'used' array is because often if a player plans the activities to great detail the node has been used several times as an identifier but because it isn't the lowest node in an activity it doesn't get assigned the activity numbers associated with those activities. If a node's 'used' component is not turned to 'used' and the activity array contains no activity numbers, the node is classed as an unplanned project component.

Duplication Check: Because the standard tree has a mono-stem structure each leaf node can only be a part of one activity. This particularly useful to check for duplication. The activity array of each node should only contain one activity for no duplication of work. The duplication check views each node's activity array, if the array contains multiple activities those activity numbers are stored and the player can view those duplicated activities during feedback.

False Activity Check: There are two types of false activities:

- Activities without enough identifiers to identify the branch
- Activities that don't match any branches in the standard tree.

The mono-stem structure of the tree can result in matching multiple nodes, each part of a different activity. In order to match the activities to their corresponding standard tree branches, each of the activity components must be correctly matched to its standard counterpart. To get a match there must be enough identifiers in the activity (The Summer-House program requires a minimum of two identifiers). If this not the case, e.g. 'cut East', the activity is stored as a false activity.

A second check to identify the lowest ranking node in the activity. It then compares it to all the nodes in the tree, once found; it compares the nodes 'above array' with the activity's second lowest ranking node for a match. If there is a match, the second lowest ranking node becomes the lowest node and the process is repeated until the whole activity is matched.

Detail Check: The game allows the 'editor' to choose a level of detail the player must plan his or her activities down to. There is a ' project detail counter' that keeps track of the level of detail of the activities planned. If at the end of all the planning the counter is less than the editor's detail level teaching objective. The player pays a penalty and during feedback can find that their plan lacked detail.

The plan is also checked for detail consistency. The 'project detail counter' represents the lowest level of detail in the plan. Each activity is then compared to this lowest level of detail. If the activity can be planned down to this level but is not, the procedure works out how far the lowest node in the activity is from the top and from the detail counter level, or from the lowest level available. This is then fed into an equation to represent a penalty. This method is used because not all activities will be represented in the standard tree down to the same level.

5. TESTING OF THE GAME

The game described in this paper is an experimental program, designed to show the theories described in the methodology are possible. The game has also many problems and limitations. To identify these problems the game was tested both technically and for player's response to the game. Five students have tested the game and their response to design and layout was very positive. Samples of the main points and responses made by the players are:

- Introduction and game instructions were clear and easy to understand
- Dictionary was very helpful. " There is nothing more frustrating than continually entering something into the computer and it is not recognising it, "the dictionary lets the player see what words they can use".
- Players were confused by words that could have multiple meanings.
- Players didn't understand the score. It didn't carry any meaning or relevance.
- Feedback screen was confusing but once players worked it out, the feedback information was helpful.
- 'More enjoyable than sitting in lectures because you can have a go in your own time and can learn more by experimenting".
- "Playing the game is like a problem solving, which is interesting and fun".

6. CONCLUSIONS

A literature study has shown that there are a variety of past and current construction management games available. Most of these games are based on good ideas and theories but most lack the computer technology of today. Also these games are designed for the player to make decisions from a set of options.

The game described in this paper is an experimental project planning game that aims to make the player realise their own options, input them and get a new resulting situation.

The game uses trees to represent and evaluate planned activities. The methodology behind this is sound, but limited. The method of mapping the trees onto the standard tree can be improved, resulting in a better evaluation of the player's performance and giving the player more freedom over his or her choices. This generation of trees can also be used in other applications, like resource planning. The game is not a complete program. It demonstrates what a complete game could be, but is not yet ready to be used as a teaching tool.

7. REFERENCES

AbouRizk S. (1994) ' Simulation and Gaming in Construction Engineering Education', ASCE Proceedings.

Au T., Parti E. (1969) ' Building Construction Games-General Descriptions', Journal of the Construction Division, ASCE, Vol. 95, pp.1-9.

Booth J., (1993) ' The Evaluation of project Models in Construction', PhD thesis submitted to the University of Nottingham.

Cullingford G., Mawdesley M., Davis P. (1979) ' Some Experiences with Computer based Games in Civil Engineering Teaching', Coputers and Education Journal, Vol. 3, Pergamon Press, pp.159-164.

Fruchter R. (1996) ' Multi-Site Cross-Disciplinary A/E/C Project based Learning', Proceedings of the Third Congress on Computing in Civil Engineering, Anheim, pp.126-132.

Halpin D. (1976) 'CONSTRUCTO-An Interactive Gaming Environment', Journal of the Construction Division, Vol.102.

Elgood C. (1989) ' Handbook of Management Games', Gower Publishing Company Ltd., UK.

Tatum C. (1987) ' Balancing Engineering and Management in Construction Education'.

Sawhney A., Mund A. (1998) ' Simulation Based Construction Management Learning System', Winter Simulation Conference Proceedings, W. Michigan University.