Proceedings of the 2013 Winter Simulation Conference
R. Pasupathy, S.-H. Kim, A. Tolk, R. Hill, and M. E. Kuhl, eds

DEVELOPMENT OF A DISTRIBUTED CONSTRUCTION PROJECT MANAGEMENT GAME WITH COTS IN THE LOOP

Yasser Mohamed

Hole School of Construction Engineering
University of Alberta
Edmonton, AB, T6G 2W2, CANADA

Mostafa Ali

Hole School of Construction Engineering
University of Alberta
Edmonton, AB, T6G 2W2, CANADA

ABSTRACT

Simulation games are effective tools for training and interactive learning but require significant development effort. A Construction project management game has been developed to provide an interactive simulation environment to assist in construction project management learning. The objective of this game is to help users understand the effects of their resource allocation decisions on project time and cost performance. A distributed approach using High Level Architecture (HLA) has been used in developing this game. The game federation consists of four federates (Progress simulator, Resources simulator, Controller, and User Interface). A Commercial-Of-The-Shelf (COTS) application "Microsoft project 2007®" is used as the main user interface for this game. The use of a common COTS reduces the effort needed for building a dedicated graphical interface from scratch and provide a familiar interface for users. This paper describes the development process of this game, potential use, and future extensions.

1 INTRODUCTION

Using games in construction project management teaching and training has been a subject of a number of research and development efforts These games try to mimic real life operations using simulation techniques and cover different construction and management aspects. According to (Gilgeous & D'Cruz, 1996) games are good media for learning because of their effectiveness in delivering concepts through application, trial and error. One of the early construction management games is Constructo (Haplin & Woodhead, 1973), which used the CYCLONE simulation language to simulate construction project execution and SuperBid (AbouRizk, 1993), which simulates construction bidding process. (AL-Jibouri & Mawdesley, 2001) developed a game to teach players about construction project planning and control through introducing them with realistic models (they used constructing a dam as an initial project but the engine can accept any kind of projects). They used Pascal to code this game. In this game, the player is responsible for the performance of the project where he has to provide enough resources to complete the project in time without cost overrun. The authors illustrated the importance of careful game design as overcomplicating or trivializing the game might hinder game objectives. (Nassar, 2002) developed a

game called ER which simulates equipment purchasing and salvaging. In this game, four teams will compete against each other, each time they have to decide either to buy new equipment or sell the existing ones, their decision will depend on the equipment age, price, and required productivity. This game, instead of building from scratch, has been built as an Add-in in Excel®.
Rojas and Mukherjee developed "Virtual Coach" (Rojas & Mukherjee, 2005) as a general purpose situational simulations environment. (Son et al., 2011) developed a 3D game for construction safety, this game aims to teach students about safety issues in a virtual construction site, the player takes the role of safety inspector and he has to identify all hazards on the virtual site. Torque 3D game engine is used in developing this game. Initial results indicated the potential usefulness of this kind of games.

Using visualization helps students and practitioners understand structural components, complex relationship between them, and sequence of construction. Many researchers tried to develop visualization engines which usually use schedules to generate 4D models. (Nikolic et al., 2011) and (Lee et al., 2011) described a virtual construction simulator using Deep Creator game engine; this simulator is used for improving student understanding for the construction projects. Unlike traditional 4D applications, this simulator does not require CPM schedule for generating 4D model; instead user creates and links 3D objects and the simulator generates the schedules.

A later implementation of another bidding game (AbouRizk et al., 2010) used HLA and distributed simulation concept to develop bidding game, in this game player can compete with virtual player for awarding projects. The virtual player can be used to improve bidding skills of the user.

This paper discusses the development of a construction management game using distributed simulation approach. The objective of this game is to engage students in resource allocation decisions and simulation their effects on project cost and schedule performance indices. Distributed simulation is used in developing this game, to enable dividing the development effort between different teams so that each team can use its preferences of simulation algorithm and programming language. In addition, using distributed simulation enables us to use existing components which have been well developed and tested for years, this will save a significant time while maintaining stability and simplicity. Microsoft Project® has been used as an interface federate in this game to give the user a well-known interface while reducing the time required to code an interface from scratch. The paper will discuss the structure of the game, the role of its different components and the implementation of Microsoft Project® as a federate. It also addresses the future development of the game.

2 GAME OVERVIEW AND STRUCTURE

There are several construction project management games that are being used for educational purposes in different institutions. A user plays the role of a decision maker (e.g. project manager) who has to take decision on the basis of existing project circumstances and the goals of the project. In this game, the initial objective is to allow the user to make resource allocation decisions while getting feedback on the effect of these decisions on project cost and schedule indexes. The game development follow a distributed simulation approach using the High Level Architecture (HLA) standards (IEEE1516). In this approach, the game represent a federation that is made of four main components (federates) that interact with each other during runtime to produce the overall game behaviors. Figure 1 shows these components in addition to other candidate components that can be added in future extensions. The main components are a Controller federate, a Player federate, a Progress Simulator, and a Resource Simulator, which simulates the performance of resources under different productivity influences. The Controller defines the project scope, work packages, schedule, work area and resources needed to complete the task. The player federate is responsible for interfacing with the user, who takes the role of the contractor's project manager and is responsible for the resource selection and allocation. The Resource Simulator initializes resources and changes their total available quantities and production rates based on player decisions and other external factors. The progress simulator is responsible for simulating the cost and schedule performance of the project in response for player decisions. The following sections describes the role of each of these components followed by their implementation with more focus on the player federate.

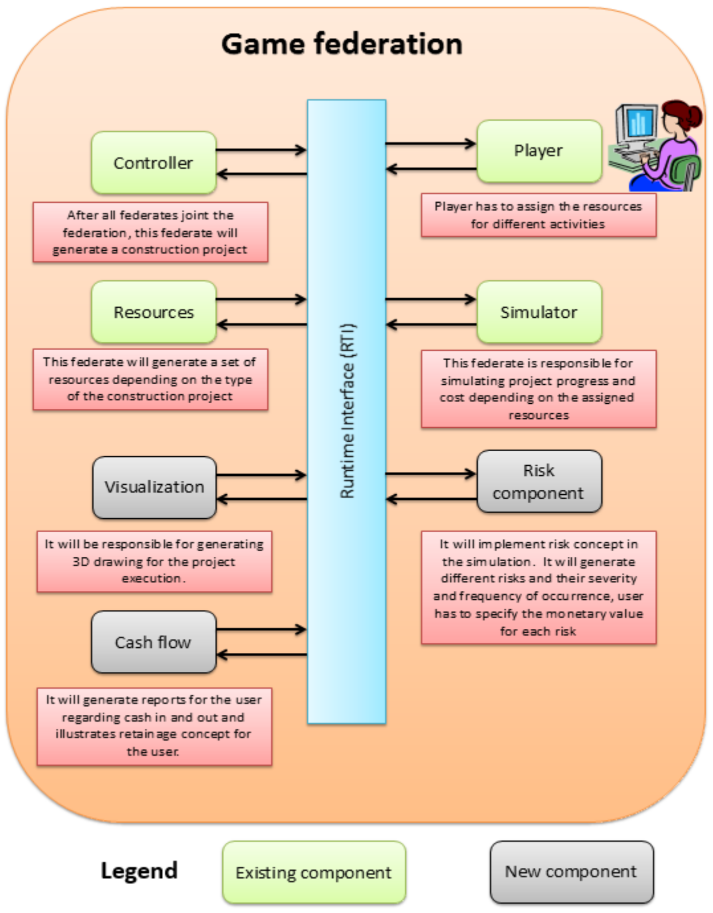

Game federation
Controller
After all federates joint the federation, this federate will generate a construction project
Resources
This federate will generate a set of resources depending on the type of the construction project
Visualization
It will be responsible for generating 3D drawing for the project execution.
Cash flow
It will generate reports for the user regarding cash in and out and illustrates retainage concept for the user.
Runtime Interface (RTI)
Player
Player has to assign the resources for different activities
Simulator
This federate is responsible for simulating project progress and cost depending on the assigned resources
Risk component
It will implement risk concept in the simulation. It will generate different risks and their severity and frequency of occurrence, user has to specify the monetary value for each risk
Legend
Existing component
New component

Figure 1. Federates in the construction game federation

2.1 Controller Federate

The first federate in the management game is the controller, this federate generates the construction project to the player. It defines work packages and their work quantity (in man hours), their resource type requirements, support resource requirements (e.g. cranes if any), and their work areas. However, it is player's responsibility for assigning resources to work packages. The federate allows for coding different types of projects. Future extensions of this federate will allow it to control some parameters of the game like resource availability scenarios, learning rates, rework, and material procurement scenarios.

2.2 Resources Federate

This federate is responsible for managing resources. At the beginning, a set of different resources, their hourly rates and production rate is populated through this federate. This set of resources can be more or less than needed by the construction project generated by the controller.

Therefore, the Resources responsibility is to:

- Create a set of initial resources that are suitable for the generated project by the controller.

- It also provides the productivity factor, availability, shift time (day / night), shift length and number of people in each crew and cost per hour associated with each resource,
- During the game execution, this federate is responsible for calculating resources cost and incurring additional fees for overtime if any,
- In addition, it will monitor each resource available quantity and control it based on any preset resource availability scenarios.

2.3 Player Federate

In Player Federate is the main user interface with the game. The main objectives of this federate are:

- Creating a user friendly environment for the user,
- Receive interactions from user and convey the interaction to other federates,
- Presenting tasks, their durations and relations, progress of game
- Communicating progress and performance indexes to the user, and
- Allowing basic functions like: request/release resources, allocate/reallocate resources to work packages, increase/decrease allocated quantities of resources to work packages, start/stop (pause) work packages, select the number of shifts and work hours per shift, and control simulation speed (pause, slow, fast).

2.4 Simulation Federate

The main goal of the Simulator federate is to show the user (player) the outcomes of his/her decisions, along with the probability that each outcome will occur. The performance of a project can be affected by many different factors. However, the first development of the game focuses only on decisions related to allocation and distribution of resources to different activities in a project.

The Simulator federate simulates the progress of th project according to resource allocations to the work packages made by the player . Work packages progresses based on a rate which is affected by how much resources are allocated to it, what productivity factor affects the production rate given different influencing factors. A work package has a work quantity that is presented in man-hours (budgeted man-hours). The budgeted man-hours for each work package are provided by the controller. Logical relations govern the dependencies between work packages. For example, a work package cannot start unless all constraints including completion of preceding packages and resource requirements are met. Therefore, for each work package to start, the Simulator checks if all the predecessors are completed and the required resource type is allocated to the work package. The player will be charged for any wrong resource allocation.

In case all the requirements are met, then the work package will get executed and the progress will be proportional to the allocated man-hours. Allocated resources man-hours are consumed based on overall productivity factor updated regularly by resource simulator. The progress is a time-stepped progress, which means the Progress Simulator will update the progress and the performance indicators of the work package in each time step of the simulation that is assumed to be on hour. Based on the amount of resource allocated and productivity factor associated with it, the progress may stop even before the work package completion point. In this case, the player will be notified by the progress report and s/he will realize that there is a need to allocate more resources to get the work done.

The direct cost of the work package execution is charged on a continuous basis based on the resource consumption and time elapsed and the cost rate provided by the Resource simulator. Overheads (e.g. supervision) is considered to be a percentage (e.g. 20%) of the direct cost and is also charged on a continuous basis based on the time elapsed. In addition, the federate includes a penalty structure for the payment. When the work package is completed, the completion date is compared against the milestone of the work package defined by the Controller and if the task is not completed on such stipulated date, late finish monetary penalty will be applied to the total cost based on the amount of delay.

3 IMPLEMENTATION

This game was developed as a class project for distributed simulation course. The class (which consists of 8 students) has been divided into four groups. Each group chose a specific federate to develop and were given the choice to use their preferred programming language for implementing the federate. Development started by a series of meetings between different groups to set the basis for the game, this included developing Federation Object Model (FOM), creating schedule for developing, and setting development milestones. IronPython® was used to develop the controller federate, C#.NET® for Simulator and Resources federates, and VB.NET® for Player federate, which was developed as MS Project® add-in. The following sections describes in more details the FOM and the player federate.

3.1 Game Federation Object Model (FOM)

According to HLA standards an FOM defines shared attributes between different federates, any attribute that will be published by a federate must be declared in the FOM. In addition, FOM states federates that will subscribe to an attribute, a run time infrastructure (RTI) application uses this to manage exchange of information and synchronization of simulation behaviors between federates.

Table 1 shows the FOM for the game, this table is divided into two parts: Object classes, and Interaction classes. There are three main classes in this model: WorkPackage, Resource, and Calendar. Workpackage class stores information about each task in the project (e.g. duration, percent complete,…etc.), while Resource represents resources and their relevant information. On the other hand, there will be one instance from Calendar class which will contain information about overall project duration.

Table 1 FOM for the game federation with federates' publish (P) and subscribe (S) roles

	Attribute/ Parameter	Type	Progress Simulator	Player	Controller	Resource Simulator
Object Classes						
Resource						
	availableQuantity	HLAinteger		S	S	P
	availability	HLABoolean		S		P
	costPerHour	HLAdouble	S	S		P
	productivityFactor	HLAsingle	S	S		P
	type	HLAstring	S	S	S	P
Resoucre.Labor						
	numPeopleInCrew	HLAinteger	S	S		P
WorkPackage						
	actualManHour	HLAsingle	P		S	
	budgetManHour	HLAsingle	S	S	P	
	cost	HLAdouble	P	S		
	deliveryDate	HLAinteger	S	S	P	
	description	HLAstring	S	S	P	
	duration	HLAinteger	S	S	P	
	percentComplete	HLAsingle	P	S		
	predecessors	HLAobjectInstanceHandle	S	S	P	
	requiredResourceType	HLAstring	S	S	P	
	workAreas	HLAinteger		S	P	

	Attribute/ Parameter	Type	Progress Simulator	Player	Controller	Resource Simulator
	Calendar					
	projectDate	HLAinteger	P	S		
	hoursPerShift	HLAshort	S	P		S
	nightShift	HLAboolean	S	P		S
	Interaction Classes					
	PauseResume					
	No Parameters		S	P	S	S
	RequestAddResource					
	type	HLAobjectIn-stanceHandle			P	S
	quantity	HLAinteger			P	S
	AllocateResourceManHours					
	resourceType	HLAobjectIn-stanceHandle	S	P		S
	allocatedManHours	HLAinteger	S	P		S
	assignedWorkPackage	HLAobjectIn-stanceHandle	S	P		S
	ProjectPerformance					
	projectCPI	HLAsingle	P	S		
	projectSPI	HLAsingle	P	S		

3.2 Player Federate

The player federate was developed by one of the student teams as a Microsoft Project® add-in to capitalize on the data structure and user interface of the program. As shown in Figure 2, the Player Federate's code consists of four main parts:

1. Local classes to represent the main FOM classes that represent different parts of the game (e.g. work packages, resources, calendar, etc.).
2. The second part deals with the connection with the RTI and related attributes.
3. This part is responsible for connecting the code with Microsoft Project 2010®'s data structure.
4. The last part contains the code for the forms found in player federate.

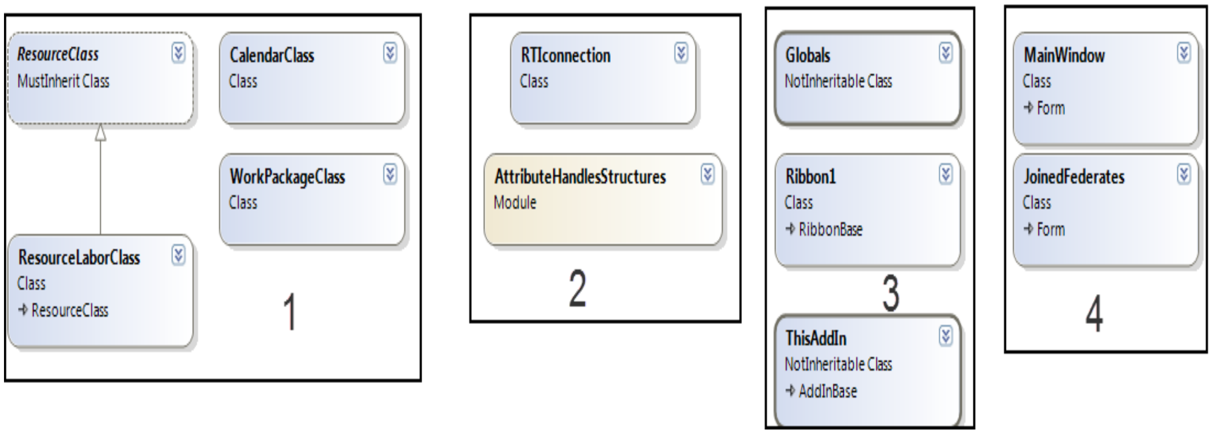


Figure 2. Main parts of the player federate

Fields, properties and methods of different classes are presented in Figure 3. Also different attributes used in this federation are shown in Figure 4.

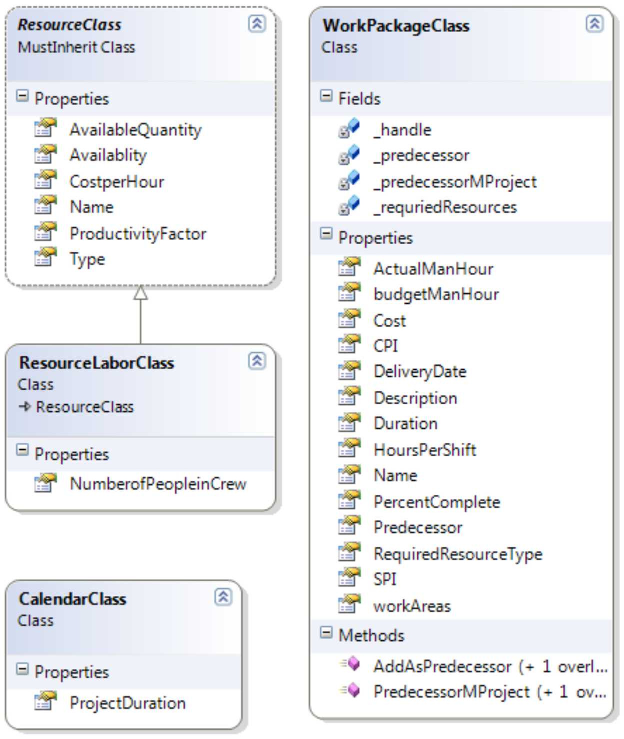


Figure 3. Fields, properties, and methods in part 1

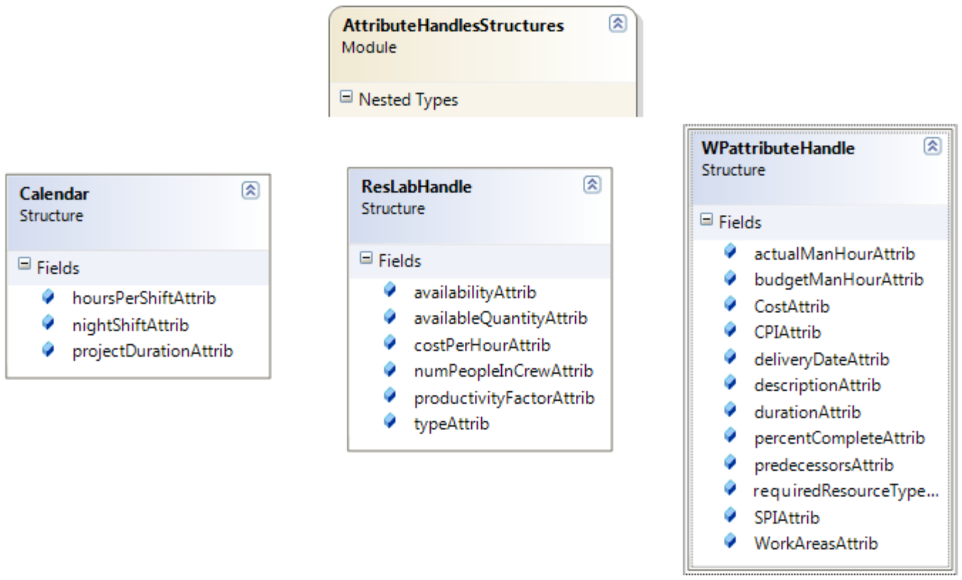
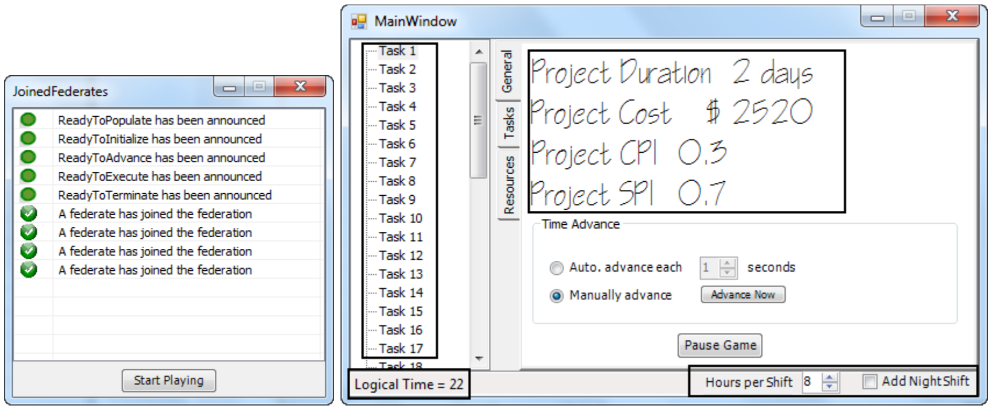


Figure 4. Different attributes used in the player federate

3.3 Interaction between Player Federate and User

3.3.1 Initialization Of Game:

On initialization of the game, the user has to run the program of Player Federate and automatically Microsoft Project will be opened as a federate. Player federate shows a window called "JoinedFederates" when the user runs Play (Management Game). After all the required federates join the federation, a button for staring the game will be visualized in this window as shown in Figure 5. By starting the game, the main window will appear, the user uses this window to assign resources and track the progress.



ReadyToPopulate has been announced
ReadyToInitialize has been announced
ReadyToAdvance has been announced
ReadyToExecute has been announced
ReadyToTerminate has been announced
A federate has joined the federation
A federate has joined the federation
A federate has joined the federation
A federate has joined the federation

Project Duration 2 days
Project Cost $ 2520
Project CPI 0.3
Project SPI 0.7

Auto. advance each 1 seconds
[✓ Manually advance]

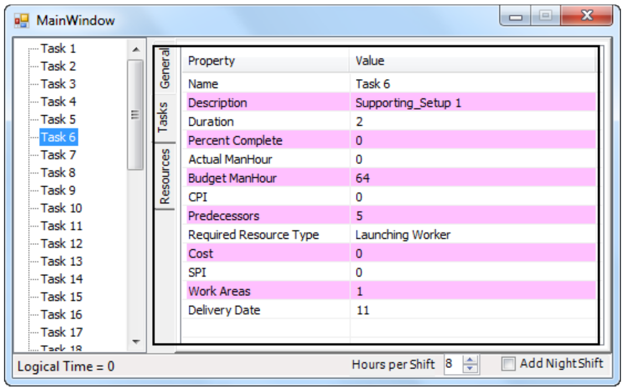
Logical Time = 22
Hours per Shift 8
Add Night Shift

Figure 5: Joined federates window and main window of the player federate

3.3.2 Resource Allocation

The user is required to make decisions on how to allocate resources to different work packages so that the target project schedule and budget are achieved. The activities for resource allocation are as follows

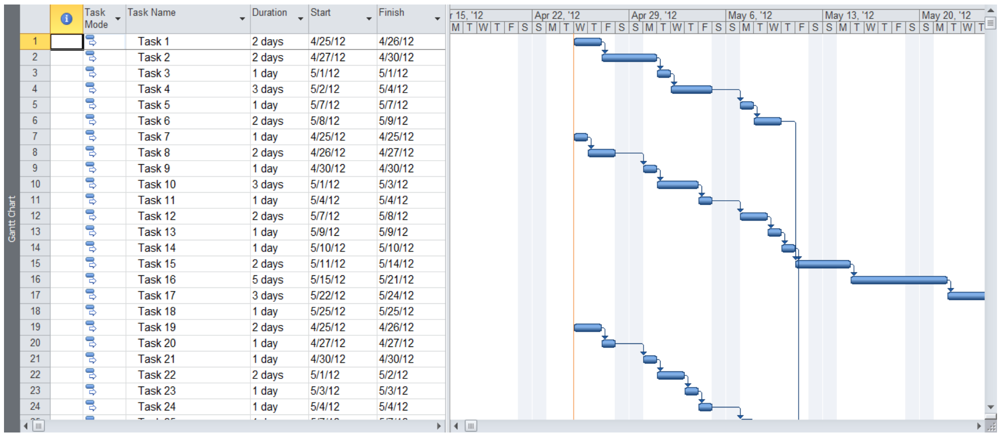
- User is informed about the required resources for each task but s/he has to make sure whether the required resources are available or not. This information is send by other federates and is made available to the user by the player federate.
- Player federate with the help of other federates also show the duration and other properties of each task with its predecessor or successor as shown in Figure 6 and 7
- User assigns required resource man hour and hours per shift to different resources as shown in Figure 8.
- If user allows night shift option then work hours per day are doubled.
- After completion of resource allocation user can start project execution.



Property	Value
Name	Task 6
Description	Supporting_Setup 1
Duration	2
Percent Complete	0
Actual ManHour	0
Budget ManHour	64
CPI	0
Predecessors	5
Required Resource Type	Launching Worker
Cost	0
SPI	0
Work Areas	1
Delivery Date	11

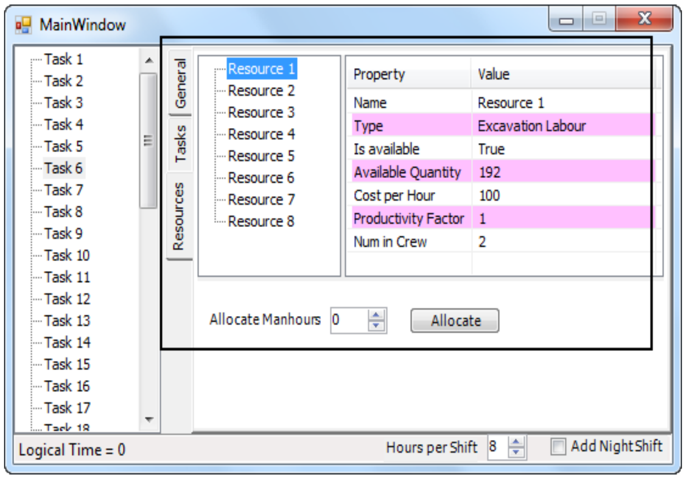
Logical Time = 0

Figure 6. Work Package (Task) properties



	Task Mode	Task Name	Duration	Start	Finish
1		Task 1	2 days	4/25/12	4/26/12
2		Task 2	2 days	4/27/12	4/30/12
3		Task 3	1 day	5/1/12	5/1/12
4		Task 4	3 days	5/2/12	5/4/12
5		Task 5	1 day	5/7/12	5/7/12
6		Task 6	2 days	5/8/12	5/9/12
7		Task 7	1 day	4/25/12	4/25/12
8		Task 8	2 days	4/26/12	4/27/12
9		Task 9	1 day	4/30/12	4/30/12
10		Task 10	3 days	5/1/12	5/3/12
11		Task 11	1 day	5/4/12	5/4/12
12		Task 12	2 days	5/7/12	5/8/12
13		Task 13	1 day	5/9/12	5/9/12
14		Task 14	1 day	5/10/12	5/10/12
15		Task 15	2 days	5/11/12	5/14/12
16		Task 16	5 days	5/15/12	5/21/12
17		Task 17	3 days	5/22/12	5/24/12
18		Task 18	1 day	5/25/12	5/25/12
19		Task 19	2 days	4/25/12	4/26/12
20		Task 20	1 day	4/27/12	4/27/12
21		Task 21	1 day	4/30/12	4/30/12
22		Task 22	2 days	5/1/12	5/2/12
23		Task 23	1 day	5/3/12	5/3/12
24		Task 24	1 day	5/4/12	5/4/12

Figure 7. Gantt chart and relations between tasks in Microsoft Project


MainWindow

Task 1, Task 2, Task 3, Task 4, Task 5, Task 6, Task 7, Task 8, Task 9, Task 10, Task 11, Task 12, Task 13, Task 14, Task 15, Task 16, Task 17, Task 18

General | Tasks | Resources

Resource 1, Resource 2, Resource 3, Resource 4, Resource 5, Resource 6, Resource 7, Resource 8

Property	Value
Name	Resource 1
Type	Excavation Labour
Is available	True
Available Quantity	192
Cost per Hour	100
Productivity Factor	1
Num in Crew	2

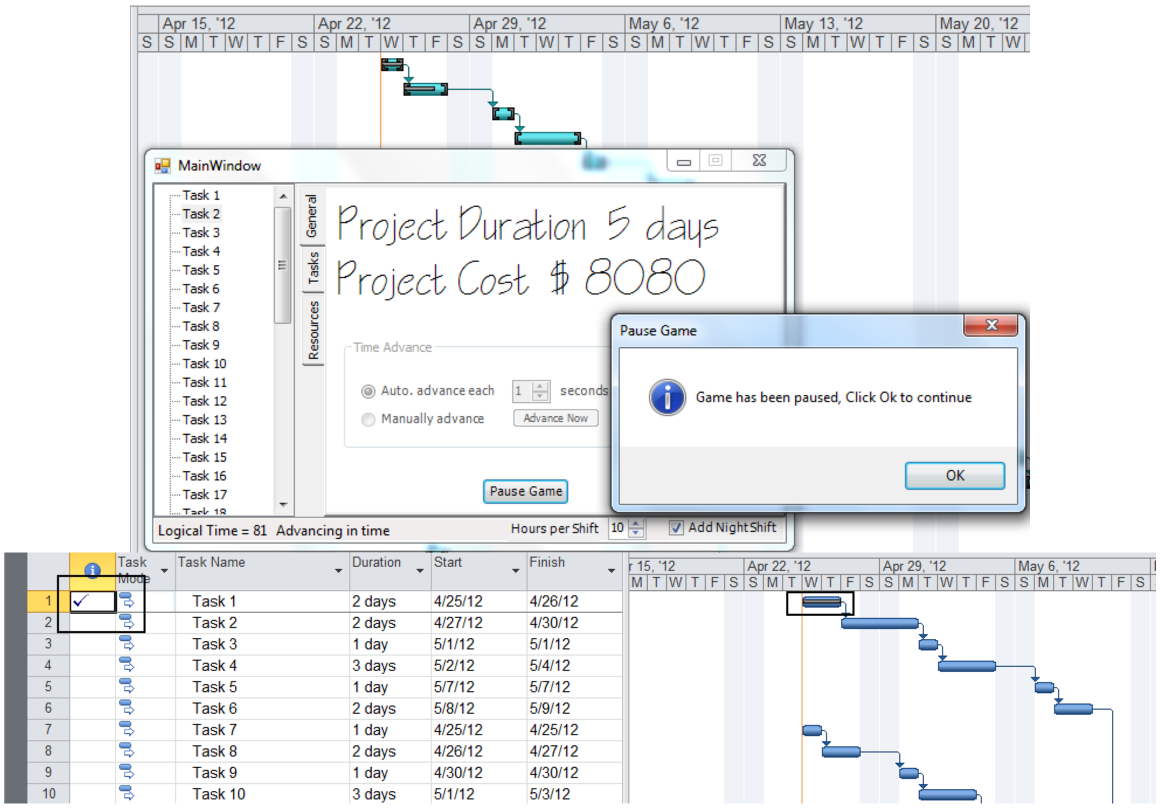
Allocate Manhours 0 | Allocate

Logical Time = 0 | Hours per Shift 8 | Add Night Shift

Figure 8: Resources allocation tab

3.3.3 Project Execution

During project execution the player federate shows the progress of the game in Microsoft Project (Figure 9). If the user thinks additional resources are required for any task, s/he can pause simulation time advance and reallocate resources. When all the tasks are being finished, the player federate shows a message about completion of project.


MainWindow

Task 1, Task 2, Task 3, Task 4, Task 5, Task 6, Task 7, Task 8, Task 9, Task 10, Task 11, Task 12, Task 13, Task 14, Task 15, Task 16, Task 17, Task 18

General | Tasks | Resources

Project Duration 5 days

Project Cost $ 8080

Time Advance

Auto. advance each 1 seconds

Manually advance | Advance Now

Pause Game

Logical Time = 81 Advancing in time | Hours per Shift 10 | Add Night Shift

Pause Game

Game has been paused, Click Ok to continue

OK

		Task Mode	Task Name	Duration	Start	Finish
1	✓		Task 1	2 days	4/25/12	4/26/12
2			Task 2	2 days	4/27/12	4/30/12
3			Task 3	1 day	5/1/12	5/1/12
4			Task 4	3 days	5/2/12	5/4/12
5			Task 5	1 day	5/7/12	5/7/12
6			Task 6	2 days	5/8/12	5/9/12
7			Task 7	1 day	4/25/12	4/25/12
8			Task 8	2 days	4/26/12	4/27/12
9			Task 9	1 day	4/30/12	4/30/12
10			Task 10	3 days	5/1/12	5/3/12

Figure 9. Game progress and task percent complete in MS Project

4 FUTURE DEVELOPMENT

Although this game development started as a course project, the successful prototype shows the potential of expanding the game into a comprehensive construction management game. Bearing in mind that the main objectives are helping students in understanding effect of their decisions and introduce an environment that tries to simulates reality, new federates can be added besides improving the existing ones.

As shown in Figure 1, a number of improvements and extensions can be made. Some of the federates that can be added to complete the game environment are Cash flow, Risk component, and Visualization.

Cash flow federate may:

- Generate financial reports for the user regarding cash in and out,
- Illustrate retainage concept for the user, and
- Show impact on project's overall financial status in case budget overrun or under run.

Risk component may:

- Implement risk scenarios in the simulation,
- Allow interaction with the user by showing him different risks and their severity and frequency of occurrence, user has to specify the monetary value for each risk, and
- Execute risks randomly during game execution.

Finally a visualization component can be responsible for generating 3D drawing for project progress.

5 CONCLUSION

In this paper, distributed simulation approach is used to develop a construction management game. The game is created based on common scheduling practices of the construction industry to allow users to examine different resource allocation strategies to maintain high project performance indices. The implementation of one federate in the game is done as an add-in to a common project scheduling application (MS Project®), which shows an advantage to the distributed simulation approach in capitalizing on existing commercial- of-the-shelf (COTS) software applications and using it as components in the game.

Future development is also discussed enhance existing components and adding new ones to add more realism to the game.

ACKNOWLEDGMENT

We would like to thank students in "CIVE 709 – Advanced Simulation Techniques" class (winter 2012) at University of Alberta for their valuable contributions in developing this game.

REFERENCES

AbouRizk, S., 1993. A stochastic simulation of construction bidding and project management. *Microcomputers in Civil Engineering,* Volume 8, pp. 343-353.

AbouRizk, S., Hague, S., Mohamed, Y. & Fayek, A. R., 2010. *An HLA-based bidding game with intelligent virtual player.* Baltimore, MD, s.n., pp. 3056-3068.

AL-Jibouri, S. H. & Mawdesley, M. J., 2001. Design and experience with a computer game for teaching construction project planning and control. *Engineering, Construction and Architectural Management,* pp. 418-427.

Gilgeous, V. & D'Cruz, M., 1996. A study of business and management games. *Management Development Review,* pp. 32-39.

Haplin, D. W. & Woodhead, R. W., 1973. *Constructo: a heuristic game for construction management.* Champaign: University of Illinois Press.

Lee, S., Nikolic, D., Messner, J. I., and Anumba, C. J., 2011. *The Development of the Virtual Construction Simulator 3: An Interactive Simulation Environment for Construction Management Education.* Miami, Florida, United States, American Society of Civil Engineers, pp. 454-461.
Nassar, K., 2002. Simulation Gaming in Construction: ER, The Equipment Replacement Game. *Journal of Construction Education,* pp. 16-30.
Nikolic, D., Jaruhar, S. & Messner, J. I., 2011. Educational Simulation in Construction: Virtual Construction Simulator1. *Journal of Computing in Civil Engineering,* p. 421–429.
Rojas, E. & Mukherjee, A., 2005. A General Purpose Situational Simulation Environment for Construction Education. *Journal of Construction Engineering and Managemen,* 131(3), pp. 319-329.
Son, J. W., Lin, K.-Y., and Rojas, E. M., 2011. *Developing and Testing a 3D Video Game for Construction Safety Education.* Miami, Florida, United States, American Society of Civil Engineers, pp. 867-874.

BIOGRAPHIES

Yasser Mohamed is an Assistant Professor in the Department of Civil and Environmental Engineering at the University of Alberta. His research interest includes modeling and simulation of construction processes and the integrating of simulation models and knowledge engineering tools into construction management and decision making processes. His e-mail address is yaly@ualberta.ca.

Mostafa Ali is a PhD candidate at the Department of Civil and Environmental Engineering at the University of Alberta. His e-mail address is MostafaAli@ualberta.ca.