Capturing Project Dynamics with a New Project Management Tool: Project Management Simulation Model (PMSM)

Ali Afsin Bulbul

Systems Science PhD Program,
Portland State University

Harder House
1604 SW 10th Ave.
Portland, OR 97201
E-mail: afsin@pdx.edu

1 Introduction

1.1 Projects often fail!

Over the last 50 years, projects have gained an important role in our lives. Much of the business world became project-oriented and organizations started to re-structure themselves in order to reach the project objectives. Moreover, management-by-projects became a widely used management phenomena, because the success of a company has never before been more dependent upon timely, low-cost project execution. Correlated to this trend, the project management area has undergone significant development. Due to its obvious benefits, many organizations and individuals have been seeking for better ways to manage projects successfully. The capability of managing projects has become a major competitive advantage for companies. Therefore, it is no surprise that the area has been extensively researched and a number of management techniques and several project management tools have been developed and applied to a variety of areas.

The extensive usage of projects fostered more research about project management. Fueled by the expansion in technological development, more sophisticated project planning and control techniques have been developed to achieve flourishing project results. But despite all these endeavors, the reality is: *projects often fail!* Schedule and cost overruns are almost the rules for the majority of projects in construction, defense, aerospace, software and many other industries (Sterman 1992). A review of 3500 projects (Morris 1987) revealed: *"Overruns are the norm, being typically between 40 and 200%."* A survey found out that less than 50% of corporate research and development projects met their objectives (Williams 2002a). A recent study showed that in a sample of ten projects, the average budget overrun was 86 % and the average schedule overrun was 55 % (Lyneis 2001).

1.2 The Problems

Why do projects often fail? Despite numerous advances in the field, problems on projects have persisted for decades. The advances in the project management techniques basically cannot cope with the complexity, systemicity, and dynamicity of the projects. Moreover, the project managers are not furnished by appropriate tools to deal with these project characteristics and to foster learning from project.

1.2.1 Project Complexity

As projects have grown larger and more complex, it has become evident that they are increasingly susceptible to cost and schedule overruns. A complex system means more than the sum of its parts and projects are characterized by complex characteristics.

According to Baccarini, *interdependency* and *differentiation* are major sources for project complexity (Baccarini 1996). He and Williams claim that the elements comprising project complexity could be considered in two dimensions: organizational complexity and technological complexity. And, these measures of interdependency and differentiation can be applied with respect to these two project dimensions (Williams 1999). These measures can be applied to operational complexity as well, where differentiation would mean the number of different tasks and interdependency would mean the degree of operational interdependencies among these tasks.

The interpretation of project complexity in this research is slightly different. In this research, project complexity is characterized by two main dimensions: *structural complexity* and *uncertainty*. Structural complexity is generated by operational and organizational complexity, and operational complexity is generated by the scope of the project, differentiation of project entities and interdependency of the project elements.

Turner and Cochrane (Turner 1993) classified project uncertainty in two dimensions: *uncertainty in the methods* (in the methods used to manage the project) and *uncertainty in the goals* (in the goals of the project). If the methods and goals are well defined in the beginning of the project, then the uncertainty of the project will be less. The uncertainty in a project can be caused by external and/or internal reasons. However, Turner and Cochrane's project complexity classification only covers internal project uncertainty. A significant source of project uncertainty is the environment. Sudden technological changes, changes in the requirements of the clients or other external stakeholders, economic catastrophes and demand fluctuations are some examples of the uncertainty from the environment. Project complexity components, their relations with each other, and their effect on the project success are depicted in the Causal Loop Diagram in Figure 1.

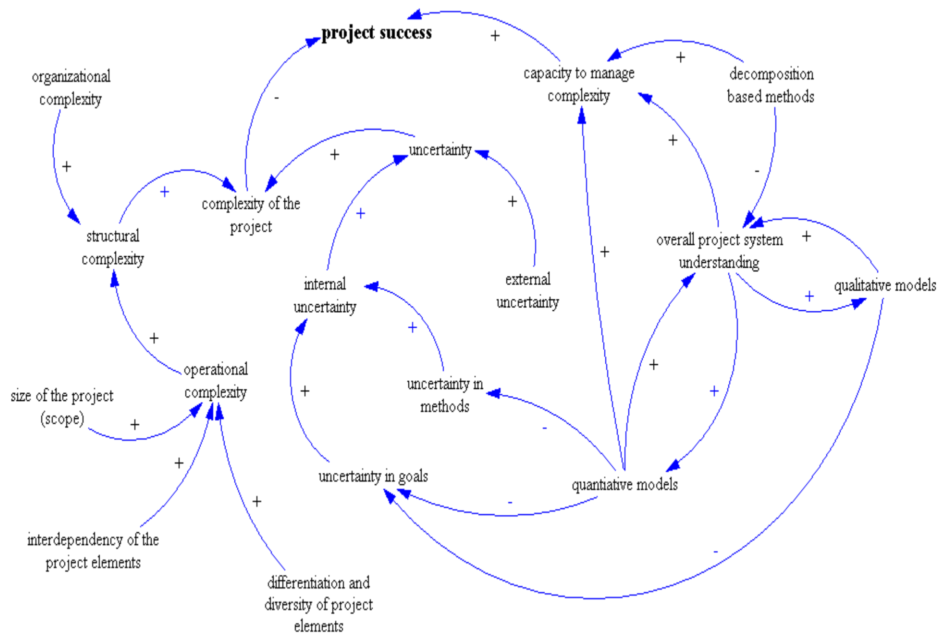
project success
organizational complexity
capacity to manage complexity
decomposition based methods
uncertainty
complexity of the project
structural complexity
internal uncertainty
external uncertainty
overall project system understanding
qualitative models
uncertainty in methods
operational complexity
size of the project (scope)
uncertainty in goals
quantiative models
interdependency of the project elements
differentiation and diversity of project elements

Figure 1 – The effects of modeling on the dynamics of project complexity

1.2.2 Holistic Nature of Projects

Most traditional project management concepts and tools utilize decomposition-based methods, which focus on the individual elements of the projects. Although these methods assist with managing *operational* entities effectively, projects usually fail because of the *strategic* project management deficiencies (Lyneis 2001). Traditional management tools offer little to reduce the potential for the overruns since they ignore the wholeness of projects.

1.2.3 Dynamicity of Projects

Both projects and project management processes are highly dynamic. According to many practitioners, practical limits of traditional project management have been achieved (Sterman 1992; Rodrigues 1998; Lyneis 2001). Although many project managers are aware of the dynamic structure of projects, traditional planning and control concepts and tools offer limited value to capture these dynamics, and they are far from satisfying expectations (Ondash 1988).

1.2.4 Learning (!) from Projects

It is quite interesting that even though project management has been used for more than 50 years, the fate of the majority of projects did not change. Overruns and actual/planned gaps are widely experienced so that in the planning phase these gaps are frequently considered normal. More interestingly, although similar projects have been executed, same old-style mistakes have been repeated continuously in organizations. In other words, organizations do not seem to learn from projects.

The growing complexity of projects has caused a dramatic increase in the need for organizations and individuals to learn from projects. Today, the capability to learn from projects can indicate the success or failure of an organization. However, most organizations still neglect the potential of learning from projects claiming: *"Projects are so unique that systematic learning cannot occur!"* This misperception is the main reason that project managers and executive management face the same *surprises* for each and every project. In most cases, project managers do not even recognize that they are repeating the same mistakes since individual and organizational learning does not occur.

Individual learning does not occur, because project managers and other people involved in projects often lack an overall understanding of the system. The effects of a management policy can be observed and memorized, but next time the same action may cause different effects because of the dynamic interdependent relationships within the project. Experience from past behaviors can be misleading, since the real understanding simply did not occur. Project managers do not have appropriate tools except some *case studies* to help them understanding the project behavior and making decisions and predictions accordingly.

Organizational learning does not occur in many organizations, because an appropriate tool to foster the learning within the organizations does not exist. For simple projects, documentation would be enough to carry out learning activity in an organization. However, in complex projects documentation cannot fully capture the dynamics of a project. Because often, the main problem is not collecting the data on what happened but it is gaining and communicating the understanding within the organization (MacMaster 2000). Traditional project management concepts do not offer effective mechanisms to facilitate the qualitative and quantitative learning. That is why, it is no surprise that for many researchers, continuous learning and improvement is placed in the highest level of project management maturity in an organization (Kerzner 2000; Schlichter 2001).

The most obvious way of learning from projects is making post-project analysis. However, the majority of the business world does not do project post-mortem analysis to examine the reasons for project success or failure and adapt their project management procedures accordingly (Williams 2002; Williams 2002a). Possible schedule overruns of the current projects increase the pressure to initiate subsequent projects leaving little (if any) time to review finished projects. However, the main reason for lack of learning is the absence of standard mechanisms and appropriate tools to promote it (Williams 2001; Howick 2002a).

1.2.5 Project Managers

Although project management techniques, concepts and tools are important to determine the success of projects, especially for complex projects, the destiny of the project is highly dependent on the capabilities of the project manager(s). A project manager is responsible for coordinating and integrating project activities across multiple functional lines. To accomplish these tasks successfully, in addition to strong communicative and interpersonal skills, and operational knowledge about the specific project tasks, an overall understanding of the dynamics of the project and experience is essential (Kerzner 1992). In many organizations, specific training is not provided to project managers. Therefore, even a novice project manager is often not given any training. Even if (s)he is lucky enough to get some training related to project management, traditional training tools are far away from providing an holistic understanding of the project, and they rarely provide a way to gain related experience. They might be helpful to improve soft skills and get necessary project information but they cannot provide *project management experience*. The only remaining way for a project manager to obtain experience is to get lessons from his (her) failures.

1.3 Purpose of the Research

To overcome these problems, a higher level, holistic perspective should be incorporated into traditional project management concepts. This research proposes a project management laboratory based on a System Dynamics (SD) model that can help to overcome the problems discussed above. SD has a huge potential to deal with the complexity of projects. It can be used as a project management and policy making tool to: a) capture the dynamics of projects, b) promote understanding, c) foster individual and organizational learning, d) help with communication among the stakeholders, and e) help project managers to make decisions.

SD project models can be successfully used to train project managers. Using an SD model as a *project management laboratory (PML)*[1], one can gain experience managing the dynamics of a project in a risk free environment. I don't argue that PMLs should (or may) substitute for the real world project management experience. However, there is no doubt that it would provide significant value.

For these purposes, the Project Management Simulation Model (PMSM) is built. The PMSM is a generic system dynamics simulation model, which can be implemented into various kinds of projects. However, it should be noted that no project management simulation model could be *ready-to-go*. It is simply impossible to build a generic model that will be used for all kinds of projects without any modification and implementation. Projects are unique endeavors, so they have some unique specifications and behaviors. On the other hand, it is true that there are some common project dynamics that are valid for almost all organizations and projects. These common behaviors are the focus of this research.

The PMSM will serve for strategic project management and it will perform:

- Project planning
    - Testing project sensitivity to risks
    - Assessing the performance of alternative policies
- Project control and monitoring

[1] The term PML is originated from the "management laboratory" concept introduced by Forester (Forrester, J. W. (1961). Industrial Dynamics. Cambridge, The MIT Press.

    - Uncovering the intangible information about the actual process
    - Helping decision making (by continuously calibrating the model with the real data from past actions)
    - Checking the performance of different project control policies
- Post-project analysis and learning from projects
- Facilitating communication among the stakeholders in a project and fostering organizational learning
- Training project managers and business students

2 Literature Review

2.1 Simulation & Systems Modeling for Project Management

With the increasing complexity of projects over time, the use of project computer simulations became a necessity. Therefore, simulation methods have been used for managing projects for a long time. Van Slyke introduced discrete simulation as a method for the analysis of project networks in 1963 (Hebert 1979). Network based techniques like CPM and PERT are analyzed by utilizing simulation methods. One of the main advantages of using simulation is that in addition to providing realistic statistics of the whole project, the data of every single project-realization is available and can be used for further analysis (Ottjes 2000). Moreover, they can reduce the risk of the project, since they provide an effective way of dealing with uncertainty (Vlatka 1998).

The idea of dynamic modeling emerged, because many project managers had recognized that the existing approaches could not deal with the direct and ripple effects (mostly caused by the changes in the requirements). So, as a complementary project management tool, dynamic project management was introduced in the mid seventies and applied in the construction industry by the Devonrue Company (Ondash 1988).

Academic work on the application of SD to project management was first developed in 1964 by Roberts to model the dynamics of R&D type projects (Rodrigues 1998). However, the real advancement in dynamic modeling of projects happened after the groundbreaking work for Ingalls Shipbuilding in the late 1970s (Cooper 1980). A SD model was used to resolve a \$500 million claim between Ingalls Shipping and the US Navy. The model was used to diagnose the causes of the cost and schedule overruns and quantify the disruption stemming from Navy-responsible delays and design changes on two multibillion-dollar shipbuilding programs (Cooper 1980; Cooper 1993b; Rodrigues 1996c; Lyneis 2001).

After this groundbreaking work, the contribution of SD models in the litigation over the cost, schedule, and scope overruns has been extensively researched and SD models have been applied to various cases. (Eden 1994; Williams 1999; Howick 2001; Williams 2001; Howick 2002; Howick 2002a). To understand the impact of disruptions and delays on a project, two methods are often used to deal with these disruptions and delays: *measured mile analysis* and *systems dynamics*. Measured mile analysis provides a benchmark for predicting the expected outturn for the project if no disruptions were to occur for the remaining part of the project through completion (Eden 2003). SD modeling seeks to replicate the impact of the streams of disruptions and delays (as categories rather then events) (Rodrigues 1996c; Howick 2001; Howick 2002).

A summary of the applications of SD to the project management area is given in Appendix.

Pugh-Roberts/PA Consulting has applied SD to more than 30 contract disputes and more than 75 different design, construction and development projects in a *proactive way* (Lyneis 2001). In addition to Pugh-Roberts/PA Consulting, many others have applied SD to improve project management both in theoretical and practical ways. Abdel-Hamid used SD models to improve managing software design projects (Abdel-Hamid 1983; Abdel-Hamid 1991). Ford, Sterman, and Repenning applied SD models for product development projects (Ford 1998); (Repenning 1999; Repenning 1999b).

It is well known that SD models cannot easily contribute to the management of projects in the operational level (Rodrigues 1996c; Rodrigues 1998; Williams 2002a). SD project models are typically highly aggregated, and the entities flowing through the model are considered to be homogenous. Also, for risk analysis purposes, deterministic structure of SD is not easily amenable to probabilistic expansion like Monte-Carlo simulation technique, although understanding the system itself is an effective way of decreasing the risk (Williams 2002a). However, Monte-Carlo simulation method also suffers from not incorporating the feedback effects in a project environment (Williams 2003). Because of these characteristics SD modeling is generally considered as a *complementary project management tool* instead of an alternative approach (PMI 2000).

There have been some efforts to integrate SD models to traditional project management techniques. The main idea is taking the operational plans from the conventional methods and using them as a base for SD models. The major work done for this kind of integration is the research of Rodrigues and Williams.

Rodrigues and Williams introduced a framework in which they constructed conceptual and analytical links with the network based models, thereby allowing SD models to be used within the project management process (Rodrigues 1996a; Rodrigues 1996b; Rodrigues 1998). With this integrated model Rodrigues and Williams showed that in addition to pre-project and post-project uses, SD models could also be used in the project execution phase to monitor and control the project process.

3 PMSM Description[2]

3.1 Introduction

PMSM consists of a SD simulation model and a graphical user interface (GUI). The model has an interface, which enables users to experiment with almost all major project parameters. Therefore the user has extensive control of the model. Graphical and numerical outputs give the user the capability to track the project progress and make the necessary changes. In this section brief information about the simulation model and the GUI will be given.

The model allows defining 2 types of employees (junior/senior) and up to10 different tasks. The user has the flexibility to define different attributes for each task and each employee. PMSM also gives the opportunity to reflect task dependencies in a project. If a task dependency is assigned to a task, this means that that task cannot start before its predecessor task is accomplished.

The unit of the task scope is *mandays* and the unit of the time is *weeks*.

To maximize the use of the model, the user is encouraged to prepare a Gantt chart for the project including the project tasks, task dependencies and task planned completion times.

To make the model structure lean and comprehensible, one-dimensional arrays are used. Dimension in this context simply means *category*. Arrayed variables can be distinguished from the others with the [Tasks] addition. For example, *task schedule pressure [Tasks]* [3]means the *task schedule pressure* is an arrayed model variable and for task number changing from 1 to 10, it shows the schedule pressure of a particular task (the *task schedule pressure [1]* is for task 1).

The variables that are specific to a task have "*task*" in their name and variables that are generic for the project have "*project*" in their name (the *task work rate*, the *initial project scope*).

3.2 PMSM Model Description

The model consists of 6 sectors:

- Human Resources Sector
- Task Execution Sector

[2] For detail information about thePMSM model and GUI description refer to Bulbul, A. (2004). Capturing Project Dynamics with a New Project Management Tool: Project Management Simulation Model. Systems Science Ph.D. Program. Portland, Portland State University.

[3] The model parameters are italicized in order to make them distinguishable.

- Productivity Sector
- Schedule Sector
- Cost Sector
- Quality Sector

3.2.1 Human Resources Sector

There are two types of employees working in a project, junior and senior employees. They have different attributes (e.g. productivity, quit rate, effect of schedule pressure, overtime exhaustion).

In the Human Resources (HR) Sector, new employees are hired; they enter a training process (if training is enabled), go through an assimilation process, quit the project (with an endogenous quit rate) or fired (exogenously by the user). Also junior employees gain experience by time and become senior after the *time to get skills to be a senior*.

Although PMSM is a continuous simulation model, some discretization is made in order to make the model better represent the reality. The *total number of juniors (seniors)*, and the *number of juniors (seniors) quit* variables are converted to integers. These integer values are used in the model for further calculations. Moreover, training is assumed to be a discrete process, so conveyors are used.

3.2.2 Project Task Execution Sector

As mentioned before, in PMSM a project can consist up to 10 tasks. Tasks can be thought as small projects, since they have different attributes (e.g. scope, error generation ratio, work rate).

Most stocks and converters in the Task Execution Sector are arrayed since many variables are task specific.

Tasks are executed based on the *task work execution rates.* The execution work rate for each task is calculated by allocating some percentage of the *effective available work rate* for that task. Similarly, review and rework work rates are calculated by allocating some percentages of the *effective available work rate.* These allocation percentages show how the employees share their time among task executing, reviewing, and reworking. These allocations can be manually entered by the user or automatically calculated by the model.

The task works that are executed enter or skip the reviewing process based on the *review ratio*. The task works, which are not reviewed, accumulate in the *Task Work not Reviewed* stock and the rest of the task works enter the reviewing process. After the reviewing process, task works follow one of three alternative paths: the task works with no errors accumulates in the *Task Work with no Errors*, the task work errors caught enter the reworking process, and the missed task work errors accumulates in the *Task Errors Missed* stock. This separation is done by the *task error generation ratio* and the *ability to catch errors.*

The task dependencies are dictated with the help of *task dependency control.* The *task in progress control* is a control mechanism, which plays a very important role in the model. It captures the information of whether or not a task should be in progress. This control mechanism is used in many parts of the model and calculated in the Task Execution Sector, using the *task dependency control* and the *task work remaining*. As explained before, a task cannot start before its predecessor task is accomplished. This is forced by this control logic.

Another important calculation made in this sector is the calculation of the *task work remaining.* The remaining task work information is used in several places in the model, including the calculation of the *task in progress control, estimated task completion times, remaining task work ratios for junior (senior) work rate allocation.*

3.2.3 Productivity Sector

Calculation of the task work rates is a critical part of the modeling effort, since the tasks are executed based on these task work rates. Work rates can be categorized into available and effective work rates. Both available and effective task work rates are calculated separately for junior and senior employees. The junior or senior available work rate for a task is the total available junior or senior work rate while that task in progress. However, the junior or senior effective work rate for a task is the junior or senior work rate, which is allocated for that task.

The calculations of available task work rates for junior and senior employees are slightly different, since some senior employees may spend some of their time training the new hired employees. The *available junior task work rate* is calculated by multiplying the *base work rate*, *junior task productivity, total number of juniors*, *weekly work time ratio*, and *daily work time ratio.* The *available senior task work rate* is calculated similarly (of course, using senior employee attributes) but the *training rate* is deducted. The amount of work rate that the trainers (*senior employees actually giving training*) spend for training is subtracted from the total work rate. This means that during the training, trainers can have limited contribution to the work rate. The more time they allocate for training (higher the *trainer time allocation %* is), the less contribution senior employees make to the work rate.

Training rate is calculated by the *number of seniors actually giving training to juniors (seniors), number of seniors, weekly work rate,* and *training time allocation %.* There is a control mechanism in PMSM to warn the user if the *training rate* is relatively high compared to the *available senior work rate.* This control mechanism is provided by the *senior work rate\training rate control.* If the *training rate* value is bigger than half of the maximum of the *available senior work rate* values, PMSM warns the user displaying: *"Your senior employers are spending most of their time on giving training! You might consider hiring new senior employees. Senior task work rate is critically low!!"*

If the *training rate* value is bigger than the maximum of the *available senior work rate* values, PMSM warns the user displaying: *"Your senior employees are spending ALL of their time giving training. They cannot find time to work for project. You might consider decreasing the training rate or disabling training."*

The effective junior and senior task work rates calculation is more complicated. The allocation of the available task work rate among the tasks can be done in two ways: equally or based on the ratio of the remaining task work for that task to the total remaining task work. General structure is as follows:

The tasks, which are executed at the same time and assigned to the same type of employees, are determined. The *available task work rate for junior (senior) employees* is shared among these tasks based on the allocation method.

In PMSM, the productivities are task and employee type specific. Although productivities of junior and senior employees are calculated separately, the structure is similar. Task productivity is a function of the expected productivity, *organizational learning coefficient*, *productivity % gain or loss due to schedule pressure*, *productivity loss due to over work* and *productivity loss due to over crowding.* The *expected productivity* and *productivity % gain or loss due to schedule pressure* is employee type specific, because junior and senior employees may have different base productivity values, and the effect of the schedule pressure on the employees depends on the experiences of the employees. The *productivity % gain or loss due to schedule pressure* is also task specific. This is trivial, since the schedule pressures for each task is calculated separately when "focus on the project milestones" project schedule management method is selected.

The *organizational learning coefficient* is a graphical user input, which reflects the effect of organizational learning during the project. The *productivity loss due to over work* is calculated based on the daily *overtime work* and *weekly work days.* The user enters the *weekly workdays* and *daily work hours* for the project. Based on these inputs, work rates and overtime work can be calculated. Normal work conditions are 5 days of work per week and 8 hours of work per day. The effect of the *daily overtime work* on productivity varies according to the number of days worked per week. The effect of daily overtime work on work force productivity is more severe if the number of days worked weekly is higher. The equations that are used in these calculations are provided by the Bureau of Labor Statistics.

If the work environment is crowded, there will be some productivity loss due to communication problems and increasing administrative work. To reflect this phenomenon *productivity loss due to over crowding* is calculated. The user enters the *normal number of employees* and the *overcrowding coefficient* is calculated, based on this information and the *total number of employees* working in the project.

The *productivity % gain or loss due to schedule pressure* is a graphical function that defines the relation between the schedule pressure and the productivity. Senior employees are more experienced, so they are less affected by the schedule pressure[4].

---

[4] Schedule pressure can be task specific or project deadline specific. Also it can be less than 0, showing that the project is expected to finish earlier (based on the current conditions). More detail is given in the Schedule Sector.

Another effect of overtime work is the *Overtime Exhaustion*. The O*vertime Exhaustion* affects the project performance in two ways. There is a certain limit for the capacity of the employees to bear severe work conditions. When the exhaustion level gets significantly high, PMSM warns the user displaying the following alert: *"Your employees are getting exhausted! Consider decreasing daily work hours and/or weekly work days."*

If the exhaustion increases more and a certain threshold is exceeded, the employers cannot tolerate the work conditions any more and the project is terminated. PMSM announces this by: *"Your employees are overwhelmed and exhausted because of extensive work conditions. Project is failed!"*

The *Overtime Exhaustion* also affects the quit rate, which is mentioned in the HR Sector Description section. The *Overtime Exhaustion* accumulates during the overtime work period (*exhausting*) and diminishes over time when work conditions are better (*de-exhausting*).

3.2.4 Schedule Sector

In the schedule sector estimated and actual completion times, task schedule slippages and project delay, and schedule pressures are calculated. The effect of schedule pressure can be task specific or project specific, based on the *project schedule management decision* made by the user. If the user prefers to manage the project by milestones, then the workforce will focus on accomplishing individual task deadlines instead of focusing on the project deadline.

The *estimated task remaining times* are calculated by the *task work remaining, effective available task work rate,* and *task in progress control.* If the task is in progress, the estimated remaining time for a task is calculated by dividing the remaining task work by the work rate of that task. If the task is accomplished then the estimated remaining time is zero, and if the task has not started yet, the *estimated task remaining time* is calculated by subtracting the *time* from the *planned task completion time*. The *estimated project completion time* is equal to the maximum value of the *estimated tasks completion times* (latest task completion time).

The *estimated task completion time* is calculated by the *estimated task remaining time* and the *actual task completion time.* If the task is accomplished and therefore the *estimated task remaining* time value is 0, the *estimated task completion time* is equal to the *Actual Task Completion Time* value. If the task has not started yet, or it is in progress, the *estimated task completion time* equals to the *estimated task remaining time* plus *time.*

The calculation of the *actual task completion time* starts only after that task has actually started. Before a task starts, the actual completion time for that task is 0. The calculation logic is complicated, but it is analogous to the time keeping process using a chronometer. When a task starts, the chronometer starts keeping track of the time until the task is finished. However, for the tasks that do not start at time 0, the start time information should be gathered and used as a base value. For example, if Task 2 is dependent on Task 1 (*task dependency [2]* = 1) then until Task 1 is finished, Task 2 does not start and therefore the *actual task completion time [2]* equals to 0. When Task 1 finishes then Task 2 starts and Task 1 actual completion time information is used to calculate the Task 2 actual completion time (as the start-up value).

Task and project schedule slippages are calculated based on the *project schedule management decision*, estimated and planned completion times. The schedule pressures are computed using these schedule slippages information.

The project completion control is a control mechanism to investigate whether or not all project tasks are finished. The *project task remaining* information is used for this control. The *project task remaining* is basically the total of the *task work remaining* values.

Similar to the *estimated project completion time,* the *project completion time* and the *project deadline* are also determined by the latest project task. The *project delay* is calculated using this information.

There are two different types of schedule pressure in the project: the *project schedule pressure* and the *tasks schedule pressure.* If the user prefers to manage the project focusing on the project milestones, then a specific schedule pressure for each task will be effective in the project. Since there will not be any schedule pressure for the tasks that are finished, the *effective task schedule pressure* is used. This effective schedule pressure is the total schedule pressure of the tasks that are in progress. On the other hand, if the focus is on the project deadline, then the effect of a generic *project schedule pressure* will be considered.

The schedule pressure differs from the schedule slippages because it has memory. The slippages should be realized as instant pictures of how the project is doing to fulfill its schedule requirements. These instant values can change abruptly with the sudden changes in the project scope or workforce level. However, in reality the effects of these changes on the performance of the workforce are not seen immediately. Therefore, in PMSM the effect of schedule changes is reflected with the schedule pressure. Schedule pressures are accumulations and modeled as stocks.

3.2.5 Cost Sector

The project cost calculated in the model is basically the cost of labor. In reality, the project cost includes many other cost sources and it is possible that in some projects labor cost might be only a small portion of the total project cost. However, in PMSM only cost of labor is considered, since a project manager would have more control on the workforce parameters than the other cost factors. Moreover, PMSM is a generic model and it is designed to capture only generic project characteristics.

In the calculation of all cost factors mentioned above, the *juniors (seniors) weekly cost* is used. These weekly costs are calculated by multiplying the values of the *number of juniors (seniors), junior (senior) hourly wage, weekly workdays, and daily work hours*.

In the model, the project cost, cost of junior and senior employees, cost of task execution, reviewing, and reworking, cost of idle junior and senior employees, junior and senior productive labor time (PLT) cost of each task, cost of junior and senior training, and cost of project delay are calculated. Moreover, the junior and senior wage ratios, which affect the junior and senior quit rates, are calculated.

In PMSM, PLT cost represents the cost of productive workforce time. Based on the user's workforce type allocation decision for each project task, employees can remain idle. In this case, it will be important to know how much money is spent for this idle time and how much is spent for the productive labor time. Therefore, PLT cost is calculated for each task and for each employee type. The *effective junior (senior) task work rate,* and *number of tasks requiring juniors (seniors) at the same time* values are used to calculate the PLT cost for junior and senior employees. The productive labor cost of each task is the sum of junior and senior PLT cost for that task. The *project cost of idle labor time* is the difference between the *project cost* and *project cost of productive labor time*.

The *cost of project delay* is one of the payoffs of not meeting project deadline. It is the total labor cost for the time between the project completion and the project deadline (for a late project). In the model structure, when the project time exceeds the project deadline the cost of project delay starts to accumulate.

Using task execution rate, reviewing rate, and reworking rate, cost of task execution, cost of reviewing, and cost of reworking are calculated for each task and the project.

Analyzing the cost sources helps the user to diagnose the source of cost factors and the payoffs of their project preferences, decisions, and actions. To enable this type of analysis, PMSM calculates a variety of cost factors and shares with the user.

3.2.6 Quality Sector

The model variables that affect the project quality are grouped in the Quality Sector.

The *error generation rate* is the key variable for the Quality Sector. Organizational learning, schedule pressure, overtime exhaustion, and expected error generation ratio values are used to calculate the error generation ratio. If the user selects *"focus on project milestones"* method, then the *task schedule pressure* values are used. On the contrary, if the user selects the *"focus on project deadline"* option then the *project schedule pressure* value is used. If tasks have specific error generation characteristics then the user sets the *error generation type* accordingly and enters the *expected task error generation ratio* values. These task specific values are used in the calculation of error generation ratio. On the other hand, if the project has a generic error generation characteristics then the user sets the *error generation type* accordingly and enters a generic *expected project error generation ratio* value. This generic error generation ratio is used in the calculation of error generation ratio.

In the model, project quality is calculated by following formula: 100 * (project work released - (total errors generated – total project rework)) / project work released

The errors are generated only in the task execution process and the *error generation rate* is calculated in the Quality Sector by multiplying the *task executing* and the *error generation ratio.*

The generated task errors accumulate in the *Task Errors Generated* stock. Using the *Task Errors Reworked* and *Task Errors Missed* values, which are calculated in the Task Execution Sector, and the *Task Errors Generated,* the *task errors missed since they are not reviewed* is calculated. As explained in detail in the Task Execution Sector Description, after execution process task works may or may not enter to the reviewing process based on the *review ratio*. The value of *review ratio* is 0 to 1 (If the *review ratio* is equal to 1, all tasks enter the reviewing process and if the review ratio is equal to 0, all task works skip the reviewing process). The task work that are not reviewed accumulate in the *Task Work not Reviewed* stock. Total amount of errors in these not-reviewed task works is the *task errors missed since they are not reviewed* value.

The *task review work rate* and *task rework rate* are calculated for each task, based on the effective available work rate, the ratio of the average review and rework times to the average task execution time, and the work rate allocation percentages for reviewing and reworking. The *review time\task execution time* is a user input that shows the ratio of average task reviewing time to average task execution time and the *rework time\task execution time* is a user input, which shows the ratio of average task reviewing time to average task execution time. For example, if the *review time\task execution time* value is 0.1, this means that executing a fixed amount of task work takes 10 times more time than reviewing it. Similarly, if the *rework time\task execution time* value is 0.5, this means that executing a fixed amount of task work takes 2 times more time than reworking it. In reality, it is likely that reviewing and reworking takes less time than task executing. This phenomenon is reflected into the model using these ratios.

Allocation preference of the available work rate among task execution, reviewing, and reworking work rates significantly affects the project performance. This allocation can be manually done by the user or automatically by PMSM. The user makes this work rate allocation decision in PMSM GUI. This decision is captured by the *work rate allocation decision* parameter.

If the user has specific preferences for this allocation then the allocation is made manually and the user enters the allocation percentages for task execution, reviewing, and reworking in the GUI. The *manual work rate allocation % for task execution, manual work rate allocation % for reviewing, and manual work rate allocation % for reworking* are the external parameters that store this information.

For a specific task, as soon as the task execution process is finished, the percentage of the work rate that is initially allocated for task execution is shared equally between the reviewing and reworking rates. To illustrate this structure using the allocation percentages above, it can be said that as soon as the task execution process finishes for a task, the work rate allocation percentage for reviewing becomes 45% and the work rate allocation percentage for reworking becomes 55%.

For example, given the following information: Work rate allocation preferences are 80% for task execution, 5% for reviewing, and 15% for reworking, *effective available task work rate* is 20 mandays/week, review time-task execution time ratio is 0.1, and rework time-task execution time ratio is 0.5. The work rates are calculated as follows:

task execution work rate = 20 mandays/week * 80% = 16 mandays/week

task review work rate = 20 mandays/week * 5% / 0.1 = 10 mandays/week
task rework rate = 20 mandays/week * 15% / 0.5 = 6 mandays/week

On the other hand, if the *"automatic work rate allocation"* option is selected in the GUI, PMSM model dynamically calculates the allocation percentages for a flawless process. The calculated allocation percentages are presented to the user in the GUI.

The automated work rate allocation percentages for reviewing and reworking are calculated based on a simple rationale. To create a smooth project process that no task work accumulates in the *Task Work to be Reviewed* and *Task Errors Caught* stocks, the amount of task work in the reviewing, and reworking processes should be same every time. Based on this rationale the following formulas are derived:

*automated work rate allocation % for reviewing = review ratio * review time\task execution time *100*

*automated work rate allocation % for reworking = review ratio * ability to catch errors * task error generation ratio * rework time\task execution time *100*

Another important structure in the Quality Sector is the work rate allocations for task reviewing and reworking. The *total task work rate* is allocated among the *task execution rate*, *task review rate*, and *task rework rate*. Since this allocation affects the project quality, this structure is placed in the Quality Sector.

Another control mechanism provided in PMSM is the *review ratio control*. The review rate control mechanism warns the user when a bad work rate allocation decision is made. If the *review ratio* is 0 then all task works that are executed skip the reviewing process and accumulate in the *Task Work not Reviewed* stock. In this situation, task reviewing and reworking processes do not occur. Thus, work rate allocation percentages for reviewing and reworking should be 0. If the user allocates the work rate manually and the work rate allocation percentages for reviewing and reworking are not 0, this will cause idle work rate. PMSM warns the user as follows: *"Since review ratio is 0, there is no reviewing and reworking process. Currently, some work rate is allocated for reviewing and/or reworking. This work rate is idle. It will be wise to set these values to 0. Or simply select automatic allocation option."*

3.3 PMSM GUI Description

PMSM GUI consists of 5 modules:

- Human Resources Management Module
- Scope Management Module
- Schedule Management Module
- Quality Management Module
- Cost Management Module

3.3.1 Human Resources Management Module

The Human Resources Management Module has navigation buttons to Initial Workforce and Employee Recruitment, Workforce Type Allocation for Individual Tasks, Employee and Organizational Characteristics submodules.

There are several graphs and numerical displays available for the users to track the progress of the project. HR related project information, such as the number of junior and senior employees, number of seniors and juniors hired, quit, and fired, number of junior and senior employees in the training process, and the overtime exhaustion warning can be monitored during the project.

3.3.1.1 Initial Workforce and Employee Recruitment Submodule

The initial workforce numbers can be entered for junior and senior workforce using the initial workforce table. These initial employees are ready to work and they do not enter the training and assimilation processes.

In PMSM, it is possible to hire or lay off employees and adjust the workforce level using the *recruitment plans*. Like many other HR-related parameters, recruitment plans can be created for both junior and senior employees. With the help of these plans, it is possible to make up to 10 different hire decisions and 5 different lay off decisions for junior and senior employees. In each decision, up to 100 employees can be hired or laid off. The employment plans are entered to a list input device. In the recruitment plans there are two tables containing *Junior (Senior) Employment* and *Junior (Senior) Lay Offs*.

The *first junior (senior) hire time* shows when the first junior (senior) employee is hired. The *number of juniors (seniors) hired 1* shows the number of junior (senior) employees hired at this time. It is similar for the second, third ...tenth hire.

Similarly the *first junior (senior) lay off time* shows when the first junior (senior) employee is laid off. The *number of juniors (seniors) laid off 1* shows the number of juniors (seniors) laid off at this time. It is similar for the second, third ... and fifth lay off.

In real life it is not so likely that an employee is recruited as soon as the hire decision is made, generally some delay is unavoidable. The *junior (senior) delay* reflects this delay and includes the time of all the administrative work done for advertising, interviews, and other necessary clerical work.

Information regarding to the usage of the GUI is provided using *information buttons*.

3.3.1.2 Workforce Type Allocation for Individual Tasks Submodule

In a project it is possible that a particular task can only be done (or preferred to be done) by only certain type of employees. PMSM lets the user reflect this phenomenon by giving the flexibility of allocating junior and/or senior employees for a particular task. These allocations are done in the

Workforce Type Allocation for Individual Tasks Submodule. Users simply click on the desired workforce type to make this allocation. Only one selection can be made for each task

3.3.1.3 Employee and Organization Characteristics Submodule

In this module, project parameters related to the characteristics of the employees and the organization are modified. These characteristics are related to assimilation, organizational learning, productivity, work conditions, crowding, and quit rates.

Some parameters are controlled by *knob input devices* (expected productivities and quit rates) and some parameters are controlled by *slider input devices* (e.g. average assimilation times, work conditions). Knob input devices cannot be changed once the simulation starts to run, but slider input devices can be changed at any time during the simulation run. Making some parameters controls knob input devices, the user is restricted to change these expected values during the project. Thus, the users are protected from being inconsistent with themselves.

3.3.1.4 Training Submodule

In this submodule junior and senior training is controlled. Using the switch buttons users can enable or disable the *junior (senior) training*. Other training related parameters such are *junior (senior) training time*, *trainer time allocation %*, and *number of seniors allocated to give training to juniors (seniors)*. The number of junior and senior employees who are in the training and assimilation processes can be monitored using the numerical displays.

3.3.2 Scope Management Module

In the Scope Management Module the parameters, which are related to task executing, are controlled. Task dependencies, initial project scope, and scope changes are defined and work rate allocation decisions are made.

Task dependency formulation works based on a simple rule and a basic assumption:

A task is independent if its dependency is equal to 0 or its task number. A task may be dependent only on one other task.

The project scope and the possible scope changes during the project are entered using the list input device. The numerical displays next to this list input device show the remaining task works.

Two types of work rate allocation decisions can be made in PMSM. The first decision is the type of work rate allocation. Task work is allocated among task executing, reviewing and reworking. This allocation can be made manually by the user or automatically by the model. If the user has specific preferences for this allocation s/he selects the *"manual work rate allocation"* option and enters the manual work rate allocation percentages.

However, PMSM can dynamically calculate the optimum allocation for a flawless process. If the *"automatic work rate allocation"* option is selected, then PMSM uses the automatically calculated allocation percentages. Moreover, the model informs the user about these dynamically changing allocation values. To see these values, users simply click on the "*click here to see the recommended work rate allocation"* button; and a new window is opened showing these recommended values. A "status indicator" is used to inform the user about the current selection. A yellow lamp indicates that automatic work rate allocation option is selected and a green lamp indicates that the manual work rate allocation option is selected.

Second decision related to work rate allocation is about the allocation of the available work rate among different tasks. When several tasks that require the same type of employees are executed at the same time, the work rate should be allocated among them. PMSM offers two different allocations: equally allocation or allocation based on the remaining task works. If the user selects the *"work rate is equally allocated"* option then the work rate is equally shared between the tasks. On the other hand, if the *"work rate is allocated based on the remaining task work"* option is selected, then the model implements an assumption. This assumption is the following: when several tasks require same type of employees and the same time, the work rate is allocated among them based on the ratio of the remaining task work for that task to the total remaining task work.

However, if the "work rate is allocated based on the remaining task work" option is selected then the model uses task work remaining information to allocate the work rate accordingly. For example, let's say that the initial scopes of Task 1, Task 2 and Task 3 are 400, 200, and 320 mandays consecutively. Based on these initial scopes (at time 0 initial scope is equal to work remaining), 400/(400+320)*100% of the available senior work rate will be allocated to Task 1 and 320/(400+320)*100% of the available senior work rate will be allocated to Task 6. Similarly 200/(200+320)*100% of the available junior task work rate will be allocated to Task 2 and 320/(200+320)*100% of the available junior work rate will be allocated to Task 6. These allocations will be calculated and updated automatically with the change in the remaining task works.

3.3.3 Schedule Management Module

PMSM gives the user the flexibility to select one of the two schedule management methods: *"focus on the project milestones"* and *"focusing on the project deadline".* Project milestones are the planned task completion times. If the user thinks that meeting specific task deadlines is more important than meeting the project deadline, then the *"focus on project milestones"* option should be selected. In that case, PMSM calculates specific task schedule slippages and specific task schedule pressures. If the *"focus on the project deadline"* option is selected, then PMSM calculates the project schedule slippage and project schedule pressure.

In the schedule management module, the user enters the planned completion time for each task. The schedule slippages are calculated by the model, using these user-input planned completion times, and endogenously calculated estimated completion times. The schedule pressure is calculated based on these schedule slippages. Both the schedule slippages and schedule pressure(s) are dynamically calculated and updated continuously.

The schedule pressure has significant effects on several project variables (e.g. productivity, exhaustion, quit rate, error generation ratio). Hence, the way the schedule is managed can affect the project performance dramatically. There are two status indicators in the Schedule Management Module in order to let the user monitor the schedule pressure.

This module is also furnished by several graphs showing project/task estimated completion times, project/task schedule slippages, and numerical displays showing the project/task completion times, project deadline, and project schedule slippage.

3.3.4 Quality Management Module

The parameters that directly affect the project quality are grouped in the quality management module. In this module, the user makes error generation type decision. If there is a generic expected error generation ratio for the project then the user changes the switch button and enters the expected error generation ratio. On the other hand, if each task has specific expected error generation ratio then the user changes the switch button accordingly and enters the task specific expected error generation ratios.

Review ratio is the ratio of the tasks to be reviewed to the task scope, and affects the quality dramatically. The *expected ability to catch errors* is another important project parameter that affects project quality. These two parameters are controlled in this module. A knob input device is used to control the expected ability to catch errors, to make sure that this parameter won't change during the project. This is important to keep the user's decision consistent; because the actual ability to catch errors is calculated using this *expected ability to catch errors* value and organizational learning curve. For example, if there is positive organizational learning, the actual ability to catch errors will increase over time, based on the expected value entered by the user in the beginning of the project. If the user could change this expected ability to catch errors value during the project, this would create inconsistency. However, review ratio is controlled by a slider input device, and therefore can be changed at any time during the project.

The ratio of reviewing time and reworking time to task executing time is used to calculate the review and rework work rates.

Several numerical displays are provided in the Quality Management Module, so that the user can monitor the quality of the project. Average error generation ratio, actual ability to catch errors, total project rework, total errors reviewed but missed, total errors missed since they are not reviewed, project quality, and project time values can be tracked from these numerical displays. The gap between the project quality and the desired project quality is calculated by PMSM and the status of the quality is displayed using the status indicator.

3.3.5 Cost Management Module

In PMSM, the only cost factor that is considered is labor cost. This cost is calculated using the junior and senior hourly wages, and the unit of these wages is US dollars per hour. The user enters these wages for junior and senior employees in the Cost Management Module. The information regarding to the average hourly wages in the sector is also entered by the user. This information is used by PMSM as a factor that affects the quit rates for employees.

In the model, several types of cost information are calculated (e.g. total project cost, cost of junior and senior employees, cost of juniors and seniors idle times, cost of project delay, cost of task executing, reviewing, and reworking, and cost of junior and seniors training.

4 Model Validation

4.1 Why Validate?

As it is widely stated among modelers, "*there is no correct model but there are useful models!*" Since all models are approximations and are simplified representations of the real world, they cannot be one hundred percent right. So one can say that no model can be ever verified or validated (Yourdon 1979; Kelton 2001).

So, why bother spending time on validation of a model? Why not spend valuable modeler time on data collection, conceptual model building, model formulation, user interface design or documentation? Because, it is often the modeler's single biggest challenge to increase the model users' confidence on the model. It is almost impossible to have a successful model without building a confidence between the user and the model.

Many times a modeler's goal is to help their clients to make better decisions, decisions informed by the best available model. The closer the model represents a real world system, the more accurate decisions can be made by the users. Therefore, increasing user confidence is often critical.

Saying that, I can conclude that modelers should build the best model available for the purpose, in light of certain practical limitations. Based on the purpose, the modeler's resources should be allocated in a balance. If the accuracy of the model is vital, then the importance of the validation will be extremely high. If the model is designed as a flight simulator, which is primarily used for training, then user interface and learning effectiveness of the model will require more attention and validation will be a secondary issue. Hence, Forrester and Senge define validation for system dynamics models as a *"process of establishing confidence in the soundness and usefulness of a model."*

4.2 Methodology

In practice, validation of a system dynamics simulation model is often the most problematic phase. It is problematic because it is frequently delayed, manipulated or even ignored. Delaying model validation to the end of the modeling stage is a major mistake that many modelers make. Instead, validation should be seen as an iterative process starting with the model-building phase, which involves clients or/and users to the process where applicable. Manipulation of validation results is also an important issue. Unfortunately, validation is sometimes designed to prove the model is right, which makes learning and progressive communication difficult. This biased validation ultimately erodes the utility of the model and the credibility of the modeler. Even worse, many times validation is ignored entirely and many important tests are simply never done.

Validation of a system dynamics model is a multi-step qualitative process. It is qualitative rather than quantitative because system dynamics is not a traditional statistical modeling technique. Its overall purpose is to analyze the underlying trends of a system and advise how different policies influence the system. Forrester and Senge state, *"There is no single test which serves to validate a system dynamics model. Rather, confidence of a system dynamics model accumulates gradually as the model passes more tests and as new point of correspondence between the model and empirical realty are identified"* (Forrester 1980).

The qualitative nature of system dynamics validation has created controversy with those familiar with other modeling techniques. Forrester and Senge state, *"The nature of system dynamics models permits many tests of model structure and behavior not possible with other types of models. Conversely, some widely used tests, such as standard statistical hypothesis test, are either inappropriate or, at best, supplementary for system dynamics models"* (Forrester 1980).

System dynamics models are not intended to predict future values or exactly match past system data. The modelers (should) strive to create a dynamic understanding of how the system behaves now and in the future. There are no prediction or confidence intervals. There is general confusion over system dynamics models because they are not stochastic in nature. Sterman states, *"System Dynamics modelers are often faulted for their reluctance to employ formal measures of goodness-of-fit when assessing the historical behavior of models. As a result, the validity of system dynamics models is often questioned even when their correspondence to historical behavior is quite good"* (Sterman 1984).

Instead, system dynamics modelers have developed a variety of specific tests in order to validate a model. System dynamics literature is full of excellent papers and book chapters on model validation. There are basically three main types of tests for system dynamics model validation; structure validation tests, behavior validation tests, and policy validation tests.

4.3 Model Testing

4.3.1 Boundary Adequacy Test

Boundary adequacy tests assess the appropriateness of the model boundary for the purpose at hand. Therefore, the first step should be the determination of model boundaries. For this purpose helpful tools include model boundary charts and subsystem diagrams (Sterman 2000). These graphs clearly depict the boundaries of the model as well as the level of aggregation.

Figure 2 shows the model boundaries for PMSM. The figure depicts the endogenous, exogenous and excluded features. One of the main purposes of boundary adequacy tests is to check whether the important concepts for the purpose of the model are endogenous or exogenous to the model. By doing this analysis, we can check whether or not there are potentially important feedback loops that are omitted from the model.

PMSM is a project management model of a single project. Therefore, the effect of competing projects in an organization is excluded from the model. The allocation of resources among different projects and the interactions of these projects in a multi-project environment are important issues. However, they are omitted from this research because the purpose of this research is different. Instead, this research focuses on fostering understanding and learning via uncovering the complex project dynamics and hence increasing project managers' capabilities. There are some project portfolio management models in the literature discussing this area. Understanding the dynamics of a single project management structure is essential and it can be the basis of any project portfolio management research.

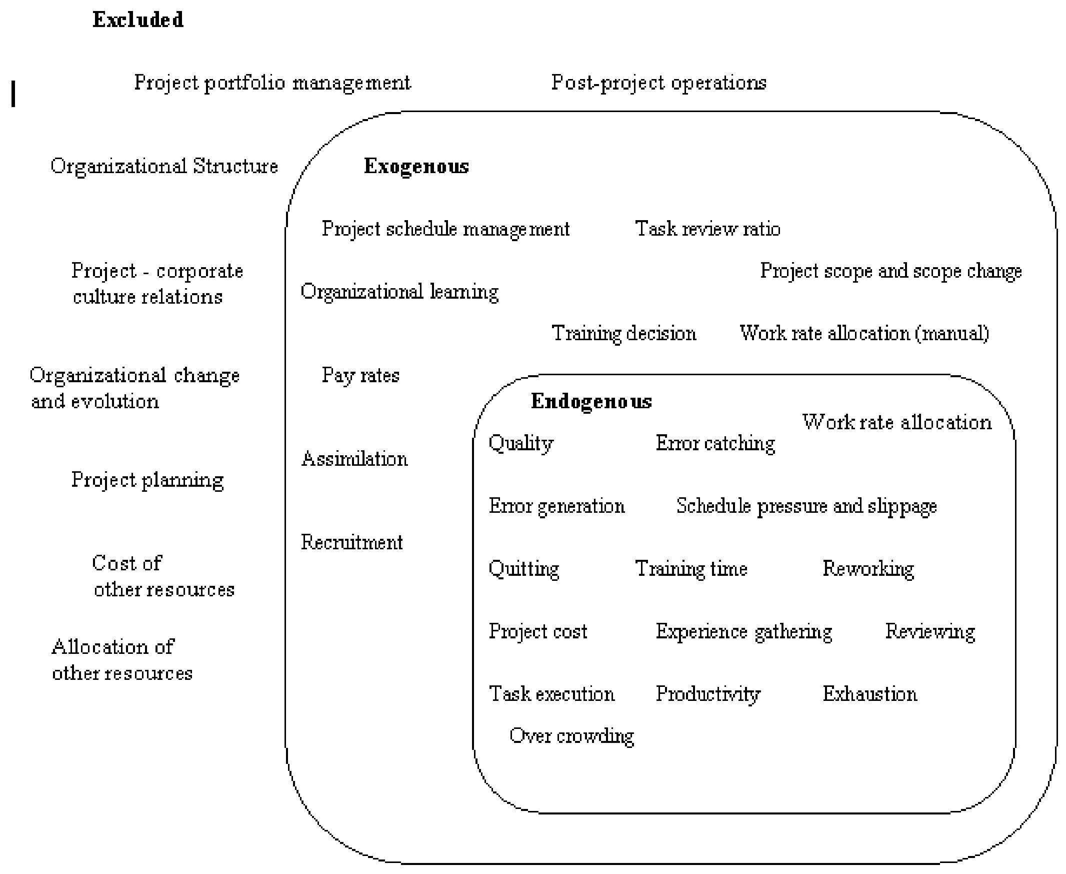

Figure 1 - Model boundary diagram

Post and pre-project activities are also excluded from the model. A project is terminated as soon as the entire project tasks are accomplished. The post-mortem project analysis, project wrap up, and documentation are not modeled. Project planning activities are also not included in the model. The model starts when project planning is already done and task execution is about to start. As a matter of fact, as discussed earlier, PMSM should be considered as a tool that can also be used during these phases. Excluding pre and pro project activities was necessary in order to be able to effectively focus on the project dynamics.

Organizational structure and the change (or evolution) of this structure are also excluded from the scope of this research. Organizational evolution is an important phenomenon, which requires special attention and detailed research. Since the main purpose of this modeling research is to focus on the basic dynamics and the analysis of the consequences of project management decisions, it was necessary to assume that organizational structure is stable and it does not affect the project management policies and project performance.

Another assumption about the model boundaries is the cost of resources. Workforce is considered to be the only project resource. Therefore, the allocation of other possible project resources is ignored, as well as the possible consequences of this allocation (e. g. competition among these resources). The main reason for not including the management of other possible resources is that they can be project specific and they may show significant dissimilarity for different type of projects. However, PMSM is a generic project management tool, which is designed to be used for all kind of project types. Moreover, it is also correct that workforce is generally the single most important resource of a project. It is controllable, and the way it is managed and allocated, many times defines the success of the project.

The number of exogenous parameters in PMSM is quite high. However, as mentioned before, model boundary analysis must be made in the highlight of model purpose. PMSM is neither a tool that is primarily designed to help the users in their decision making process, nor a model that captures project dynamics and gives users the optimized solutions to answers to their management problems. Nevertheless, it is a tool that is mainly designed to train (potential) project managers, showing the basic dynamics of the project management. It is a project management laboratory where learning is established by letting the user play with the model parameters, testing different project management policies and learning from the consequences of his/her decisions.

The exogenous parameters used in the model are capitalized. Many of the exogenous parameters are some "*base*" or "*expected*" values, which are still part of some feedback loops.

As a conclusion, keeping the purpose of the model in mind, the model boundaries are appropriate. The model is designed so that its boundaries are wide enough to cover all the major behaviors that are critical for the success of a project, but it is still a generic project management tool that is suitable for a variety of different project types. The boundaries are wide enough to make it a project tool that ensures the integrity of the project and still have enough detail to give important specific information about the main dynamic behaviors in the project.

4.3.2 Structure Assessment Test

Structure assessment tests check whether the model is consistent with the knowledge of the real system relevant to the purpose. "*Structure assessment focuses on the level of aggregation, the conformance of the model to basic physical realities such as conservation laws, and the realism of the decision rules for the agents*" (Sterman 2000). Structure assessment tests are carried out using many of the same tools useful in boundary adequacy tests, like subsystem diagrams as well as some others like causal diagrams and stock and flow maps.

Similar to many other validation tests, structure assessment tests should be performed continuously during the modeling phase. As discussed earlier, one of the most common modeler mistakes is the postponing of the validation until the model is built.

In the modeling process of PMSM structure assessment tests are done in an extremely strict fashion. The general model structure and the relationships among the different sectors in the model are studied carefully. Moreover, stock and flow structures for each sector are also carefully analyzed to find the best structure that models the project management process with the desired level of aggregation, considering the model purpose.

4.3.2.1 General Structure

The general structure of PMSM can be seen in the Model Subsystems Diagram (Figure 3). If we look closely at the relations between these sectors we can see that the following parameters are exchanged among these sectors:

The number of employees working in the project is calculated in HR Sector using the initial number of workforce, recruitment decisions, and quits. This information and training work rate information are passed to the Cost Sector to calculate the cost of the project and training. The training work rate information is passed to Productivity Sector, so that the effective senior employee work rate is calculated (by deducting the training work rate from the available senior work rate). The schedule pressure information, which affects quit rate, is received from the Schedule Sector and the average effective junior work rate is received from the Productivity Sector. The average effective junior work rate information is used to calculate the experience gathering. The more a junior employee works, the faster s/he gains experience and the less time it takes for him/her to be promoted as a senior employee.

The effect of schedule pressure in a project is enormous and it is reflected on PMSM in detail. The schedule pressure affects productivity, quality and quit rate dramatically.

In the Quality Sector the work rate is allocated between the task executing, reviewing and reworking. This allocation is a key project management policy and has dramatic effect on the schedule, cost, and quality performance of the project.

Task execution related parameters are the primary source for calculating the schedule slippage (and hence the schedule pressure because of this slippage). Errors generated, and tasks executed without error determine the quality and the task work rate is used in productivity and cost calculations.

Studying the main relations among the sectors of the model is not enough to guarantee the validity of the general structure. It is also necessary to analyze the nonexistent relations between these sectors! This way, it will give the modeler the opportunity to question some major assumptions. If there are any important feedback loops, which are omitted by mistake, the structure should be revised. This kind of high-level analysis is quite important for the structural validity of the overall system.

For example, as seen in Figure 3, there is no relationship between the HR and Quality sectors. This means any policy that is related to human resources management has no direct effect on the project quality and vice versa. It is clear that the quality of a project has no direct impact on the management of the human resources. One might think that human resources related policies should affect project quality. However, this effect is rather indirect and it is captured by PMSM; the number of employees affects the work rate and therefore affects the project schedule. A change in the project schedule creates schedule pressure (positive or negative), which affects error generation ratio. And as discussed earlier in detail, the error generation ratio has a huge impact on the project quality.

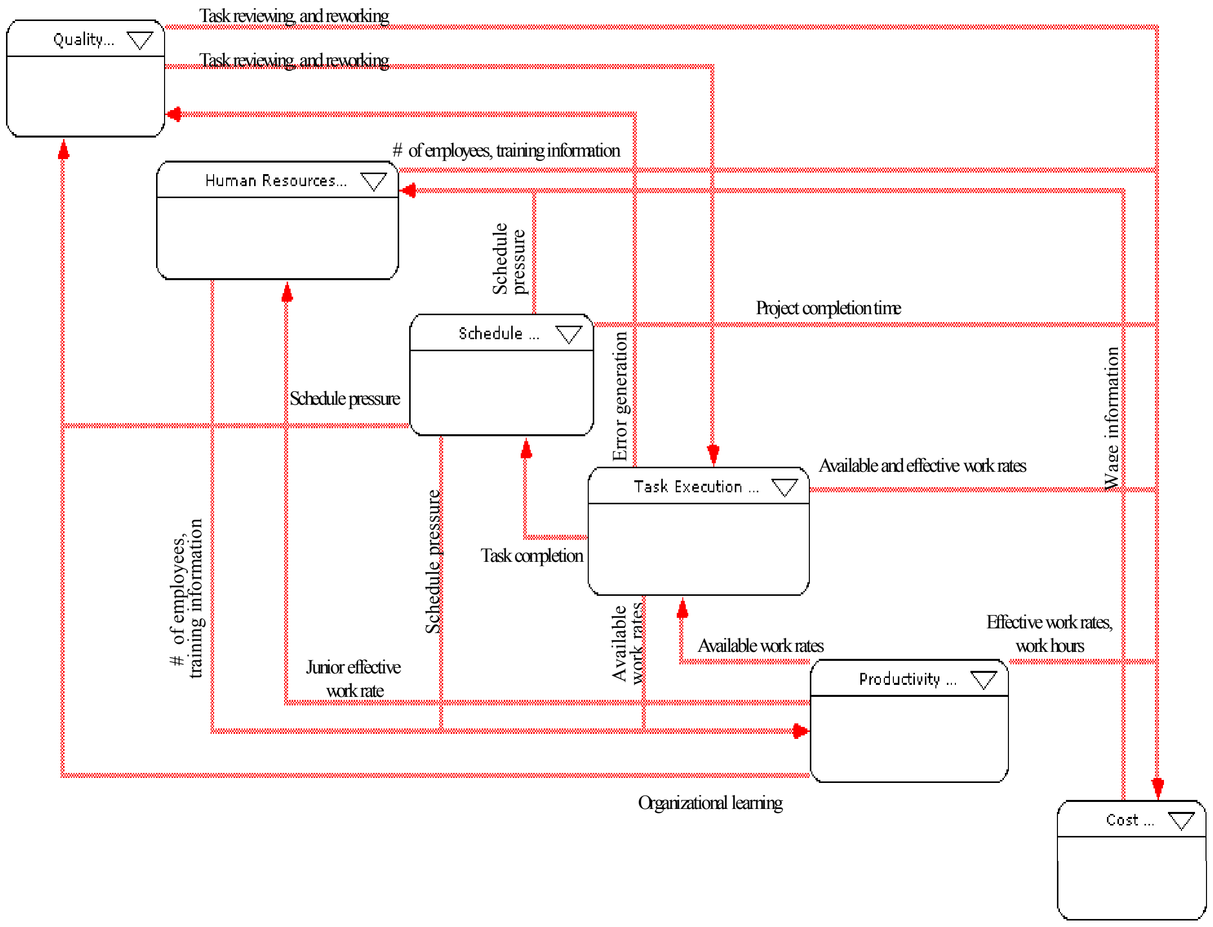

Figure 3 - Model Subsystems Diagram – Model Structure

A possible direct effect of human resources policies on quality might have been the training. When the training is enabled, it is likely that error generation ratio and/or error-catching ability might change. Furthermore, it would be a fair assumption to say that the longer the training time is, the higher the error-catching rate will be and the less errors employees are likely to make. Although training might have a significant impact on the quality, in PMSM, this effect is embedded in the effect of organizational learning. The user should define the organizational learning coefficient accordingly. In this way all the effect due to learning is captured in one parameter (otherwise training will have an learning effect as well as organizational learning coefficient, which will create confusion) and consistency of the model should be pertained.

Similarly, HR decisions have no direct effect on Task Execution and Schedule Sectors. However, there are indirect effects via productivity sector.

Since PMSM is designed as a management laboratory some loops are deliberately left open. For example, quality and cost are important project performance indicators, which vary based on the different project management policies. The model does not make any policy changes or adjustments automatically based on the changes in these indicators. In fact, the user should analyze the results after every game interval and make appropriate adjustments. In this way, the user learns from the consequences of his/her project management preferences.

4.3.2.2 Detailed Structure

During the modeling process, as a part of structure validation tests, the stock and flow structure of each sector is examined carefully, looking at the level of aggregation, closeness to the real system, existence of free lunches, inconsistencies, and the conformance of the model to basic physical realities (such as conservation laws). The final model structure is a result of various tests and experiments done continuously.

Free lunches arise when activities that require resources occur without those resources in the model. For example, this would mean that training would occur without trainers or trainees, or there would be some work done even when there are no employees working in the project. The model is checked against this kind of free lunches and existing free lunch problems are solved.

Another important structure validation test is ensuring the model conformance to basic physical realities. This means that the units in the model should be realistic, the flows should be rates, stocks should be things that can really accumulate or integrate in the real life, and conservation of matter or energy should not be violated. The units of the variables, stocks and flows should be realistic; meaning that they should have meanings in the real life.

Conservation of matter or energy usually arises because the model does not appropriately capture the stock and flow structure of the system (Sterman 2001). To ensure that the matter or energy is conserved in a system dynamics model, one must check whether the stocks, which consist of real quantities, are going negative or not. The best way to verify this is to check in extreme conditions. An example of an extreme condition is forcing the model to implausible recruitment actions (e.g. trying to fire employees when there are no employees in the system).

4.3.3 Unit Consistency Check

Dimensional consistency tests are one of the basic tests that should be done continuously in the modeling process. For each variable that is used in the model a meaningful unit should be assigned and consistency of these should be maintained. Although dimensional consistency may reveal nothing more than a typo, a missing ratio or time constant, more often unit errors reveal important flaws in the understanding of the structure or decision process (Sterman 2001).

As with many other system dynamics modeling software, Stella 8 has also an automated unit consistency check. However, it is always a better practice to check each parameter and equation in the model manually during the modeling process. Because, this automated tools can generate error messages where an equation is dimensionally consistent or they can skip some unit inconsistencies.

PMSM is dimensionally consistent. Unfortunately, due to some software constraints, Stella 8 cannot verify unit consistency for some equations.

4.3.4 Parameter Assessment Tests

Every parameter and variable in a system dynamics model should have a clear real-life meaning. After this is granted, the modeler should estimate the value of each parameter. The basic choice for parameter estimation is formal statistical estimation from numerical data, or judgmental estimation.

Since PMSM is not implemented to a real life project, there is no numerical data available. Therefore, no numerical parameter estimation is done. Instead, judgmental estimations are done that replicate well-known project behaviors documented in the literature.

4.3.5 Integration Error Tests

As mentioned earlier system dynamics models are formulated in continuous time and solved by numerical integrations. It must be ensured that the numerical integration method and time step (or occasionally named as *dT*) should be selected appropriately.

There are several parameters in PMSM that are user inputs. Some of them are related to time delays in the model like training time, assimilation time and time to get skills to be a senior. Since the modeler has no control on the duration of these exogenous delays, it is always a better practice to be cautious. Considering that the time unit of the model is weeks it can be assumed that none of the delays in the model will be less than half a day (4 hours). Assuming that a week consists an average of 40 work hours and taking one-fourth of this smallest time constant, the time step of the model can be calculated as 0.025. This value also shows the characteristics of the well-known step time selection guidelines.

A widely-used test to check the adequacy of the time step chosen is a simple test that is done by cutting the time step in half. In this test the time step is cut to half and run again. The results are compared with the original values. If there is no significant difference then the original value is fine. The time step of the model passed this test.

Euler integration method is used in the model, as suggested in the guidelines given above.

However, higher-order integration methods Range-Kutta (second and fourth order to be exact) are also used in order to check if there are significant differences. Since there was no difference, Euler is selected.

4.3.6 Extreme Conditions Tests

A system dynamics model should be robust with respect to extreme conditions as well. No matter how extreme and unlikely the conditions or policies are to occur, the model should behave in a realistic fashion. With the increasing number of user inputs it becomes increasingly more important that the model is checked against extreme conditions.

Several extreme conditions are imposed in PMSM in order to check the model. Some examples are:

- When there are no employees working in the project, obviously no project task should be done. Moreover, when there are no employees in the project, there cannot be any quits and lay offs (although might be imposed by the user).
- Although the training is enabled, when there are no senior employees assigned to give training or when senior employees allocate 0% of their time, there should not be any training (training process should be skipped).
- The number of lay offs at a certain time cannot be more than the current number of employees. Even if the user forces the model to do so model should behave reasonable.
- Even if the planned task or project completion times are extreme (0 or infinitive), the model should behave rationally. The planned completion times are used to calculate the schedule pressure, which affects the employee productivity. Extreme schedule pressure conditions (either extremely tight or loose) should not lead to unreasonable results.
- When the error generation ratio is in extreme values (0 or 1), the model should behave rationally.
- When the review ratio is in extreme values (0 or 1), the model should behave rationally.
- Model should produce meaningful results even in extreme work conditions (e.g. 7 days a week, 15 hours a day, etc.).

4.3.7 Behavior Anomaly Tests

Behavior anomaly tests examine the importance of the relationships in the model by asking whether anomalous behavior arises when a relationship is deleted or modified. These tests are extremely helpful especially in complex models like PMSM since they let the modeler to evaluate the importance of the relationships. It is a neat way to simplify the model without sacrificing the key relationships and behaviors.

Several behavior anomaly tests are executed to make sure to keep only the significant relationships in PMSM and keep the model structure as lean as possible.

Behavior anomaly tests can be executed for the relationships in a feedback loop (then also called *loop knockout analysis*) or for the relations, which are not part of a feedback loop.

An example of the loop knockout analysis done for the purposes mentioned above is breaking some loops that include the effect of schedule pressure. This gives an opportunity to check the significance of schedule pressure on work productivity, error generation, and quit rate. The effect of exhaustion due to overwork on quit rate and error generation, the effect of overcrowding, and several other important relationships are also checked.

4.3.8 Family Member Test

The family member test assesses the model's ability to generate the behavior of other instances in the same class as the system the model built to mimic. This test is particularly important for PMSM, because one of the main differences of this research from the existing ones is that it is generic to any project management process. It can be implemented to a variety of different types of projects, from software lifecycle projects to construction projects.

"You should be suspicious of any model that can exhibit only a single mode of behavior", states Sterman (Sterman 2001). PMSM can create different project behaviors depending on the input parameters selected. The model responses to different project management policies (e.g. resource allocation, desired quality, employment), creating dynamic behaviors accordingly.

4.3.9 Sensitivity Analysis

Sensitivity analysis is used to determine how sensitive is a model to the changes in the value of the parameters, the structure, and the assumptions of the model. Parameter sensitivity is usually performed as series of tests in which the modeler sets different parameter values to see how a change in the parameter causes a change in the dynamic behavior of the model. Parameter sensitivity helps to build confidence in the model by studying the uncertainties that are associated with the parameters in the models. Many parameters in system dynamics models represent quantities that are very difficult, or even impossible to measure accurately in the real world. Therefore, when building a system dynamics model, the modeler is usually at least somewhat uncertain about the parameter values. Sensitivity analysis allows the modeler to determine what level of accuracy is necessary for a parameter to make it sufficiently useful and valid for the desired model purpose. If the tests reveal that the model is insensitive, then it may be possible to use an estimate rather than a value with greater precision.

The sensitivity can be numerical, behavioral mode or policy sensitivity. Numerical sensitivity occurs when a change in a parameter causes changes in the numerical values of the model. All models exhibit numerical sensitivity. Behavioral mode sensitivity exists when the patterns of behavior generated by the model change and policy sensitivity occurs when the desirability of a policy changes with the change in the parameter.

Structural sensitivity analysis is a good way to check the appropriateness of the structure and to uncover the uncertainties that might occur due to the structure of the model. It should be incorporated with boundary adequacy and structure validation tests.

Another use of sensitivity analysis (particularly useful for complex models) is the assessment of the model results' sensitivity to the uncertainty of the modeler's assumptions. The major assumptions of the model that have high uncertainty should be analyzed to uncover their impacts. Any assumption having high uncertainty and much impact on the model behavior deserves more attention and should be verified.

In PMSM, for each important relationship and major assumption, sensitivity analysis is made. In complex models like PMSM, it is extremely hard to make parameter sensitivity analysis, since there are so many complex relations embedded. To overcome this problem, one should follow an organized process, isolating the variation in the other parameters. For each relationship, the sensitivity analysis should be done iteratively changing one parameter at a time. Only after making these single parameter sensitivity analysis, one should start to check the sensitivity of the general model behavior to the uncertainty in the selected parameters.

This type of single-parameter sensitivity analysis are useful to understand the dynamics of the relations in the model and they are performed for all relations in the model. However, the sensitivities of the model's performance indicators (quality, cost and schedule) should also be tested.

In the sensitivity analysis, the variables that are described with a single numerical value at any time (e.g. hiring delay) and more complex variables such as task dependencies are investigated. Because of the model's size, it is not possible to include all the parameters in this analysis. Parameters are eliminated if another parameter could be used to test the sensitivity of its feedback loop (e.g. *number of senior employees allocated to give training* and *training allocation percentage*). Also only one of the similar parameters is used (e.g. *initial number of seniors* instead of *initial number juniors*).

First, plausible ranges for the selected parameters are defined. And then sensitivity is tested by observing project performance across these ranges of parameter values.

4.4 Summary of Model Testing

No single test is adequate to validate a system dynamics model and no model can be validated thoroughly. A modeler should select the proper validation tests and execute them systematically starting early in the conceptual model-building phase and continuing through the whole modeling process. The validation methodology is discussed in this section. In this thesis research, a wide range of tests are performed to investigate the robustness and limitations of the model. Some examples of these tests are presented. The modeler's confidence in the model increased significantly as a result of performing these tests.

5 Conclusions

5.1 Recapitulation

Over the last 50 years, projects have gained an important role in our lives. Much of the business world has become project-oriented, and organizations often re-structure themselves in order to better achieve project objectives. The ability to effectively manage projects has become a major competitive advantage for some companies. Correlated to this trend, the project management discipline has undergone significant development.

However, despite numerous advances in the field, problems on projects have persisted for decades. Today, the advances in the project management techniques basically cannot cope with the complexity, systemicity, and dynamicity of the projects. Moreover, project managers do not have appropriate tools to deal with these project characteristics and foster learning from projects. Such tools can also help to create a common language between the stakeholders, improve their understanding, and foster communication.

To overcome these problems, a higher level, holistic perspective should be incorporated into traditional project management concepts. This research develops a project management laboratory (named as PMSM) based on a generic System Dynamics (SD) simulation model that can help to overcome these problems. PMSM can be used to: a) capture the dynamics of projects, b) promote understanding, c) foster individual and organizational learning, d) help with communication among the stakeholders, e) help project managers to make decisions, and f) train (novice) project managers and business administration students.

5.2 Major Contributions

This research provides a generic project management simulation tool with the potential to be used for planning, executing, and conducting a post-mortem analysis of projects, and for training (potential) project managers. PMSM is the first generic SD project management model to provide all of the following:

- Ability to divide project scope into tasks, of which may have different characteristics (e.g. scope, schedule, error generation, productivity).
- Incorporates the effect of dependencies among the project tasks in order to overcome the over-aggregation problem of the previous system dynamics project management models.
- Provides a graphical user interface that covers all major project management controls. PMSM GUI is easy to navigate and user friendly. Moreover, its interactive design creates a project management laboratory that fosters learning.
- Defines 2 types of employees (junior/senior) with different characteristics (e.g. productivity, quit rates, effect of schedule pressure, effect of overtime working).
- Allows the user to make two types of work force allocation decisions. PMSM gives the opportunity to allocate the available resources rate among the tasks that are being performed in parallel. The available work rate can be allocated equally or based on remaining task work ratios of the tasks that are to be performed at the same time. Moreover, PMSM allows the allocation of resources among task executing, reviewing, and reworking. This allocation can be done based on the user preferences; or the best allocation to create a *"flawless process"* may be automatically calculated by the model.
- Enables the user to investigate the effect of the schedule pressure and overtime exhaustion on the project performance. The effect of the schedule pressure is reflected into the model in detail. Furthermore, PMSM lets the user analyze the effect of two different project schedule management methods: "focus on the project deadline" and "focus on the project milestones".

- Provides a unique quality and cost structure, and a detailed reporting system that allows the user to analyze the sources of the problems that cause project quality deviations and cost overruns.

5.3 Limitations and Future Research

The model has certain limitations. The major limitations are:

- **Generic structure:** The model is designed as a generic project management tool that captures only the most common project behaviors. However, the variety of projects will always require modifying the structure to adequately reflect any specific project. These modifications include calibration of the simulation model as well as the adjustment of the GUI. This might require intensive data collection and parameter estimation processes. Formal procedures should be established to perform these processes effectively.
- **Number of project tasks:** One of the major limitations of the model is that it can contain up to only 10 project tasks[5]. Although sometimes this might be a simple aggregation issue, more often it would limit the usage of the model for complex projects without making proper changes in the model. The model should be improved to flexibly incorporate a "user-defined" number of tasks to the project structure.
- **Model size and complexity:** The size and the complexity of the model would have to increase significantly for the model to be applied to larger projects and the design of the model may not scale well.
- **Task dependencies:** Although the effect of task dependencies is uniquely incorporated to the model, there are some limitations. A task can only be dependent on only one other task in PMSM. However, in reality it is possible that tasks can be dependent on more than one task.
- **Integration to traditional methods:** PMSM can successfully serve as a complementary tool for project management. However, integration to current project management methods is essential. Moreover, standard techniques for automatic data transfer from widely-used project management software will help to decrease the effort needed to modify the model for specific projects.
- **Validation of GUI:** Although several validation tests are performed in order to increase confidence in the model, the teaching effectiveness and learning efficiency of PMSM graphical user interface has not yet been investigated.

This paper gives a brief summary of an on-going research. More detail information can be found in (Bulbul 2004). An example application of PMSM is discussed in (Bulbul 2005).

This research is a part of a PhD dissertation. In the research, so far the model development phase is accomplished and various tests are applied to the model to increase the confidence to the

[5] However, the model is designed such that this number can be expanded without changing the model structure.

model. In the next step PMSM will be applied to some real-world project cases. This will give opportunity to better validate the model and the GUI. Once this accomplished, certain characteristic project management problems will be analyzed using this new tool.

References

Abdel-Hamid, T. K., and S.E. Madnick (1983). "The dynamics of software project scheduling." Communications of the ACM **26**(5): 340-346.

Abdel-Hamid, T. K., S.E. Madnick (1991). Software Project Dynamics: An Integrated Approach. New Jersey, Prentice Halls.

Baccarini, D. (1996). "The Concept of Project Complexity." International Journal Of Project Management **14**(4): 201-204.

Bulbul, A. (2004). Capturing Project Dynamics with a New Project Management Tool: Project Management Simulation Model. Systems Science Ph.D. Program. Portland, Portland State University.

Bulbul, A. A. (2005). Intoducing a new project management tool: Project Management Simulation Model (PMSM). To be presented in Portland International Conference on Management of Engineering and Technology, Portland.

Cooper, K. G. (1980). "Naval ship production: a claim settled and a framework built." Interfaces **10**(6): 20-36.

Cooper, K. G. (1993b). "The rework cycle: why are projects mismanaged." PM Network Magazine(February): 5-7.

Eden, C. (1994). "Cognitive mapping and problem structuring for system dynamics model building." System Dynamics Review **10**(2-3): 257-276.

Eden, C. W., T. and Ackermann F. (2003). Issues with the measured mile analysis. Glasgow, Strathclyde Business School.

Ford, D. N., and J. Sterman (1998). "Dynamic modeling of product development processes." System Dynamics Review **14**(1): 31-68.

Forrester, J. W. (1961). Industrial Dynamics. Cambridge, The MIT Press.

Forrester, J. W., P. Senge (1980). Tests for building confidence in system dynamics models. New York.

Hebert, J. E. (1979). Applications of Simulation in Project Management. Winter Simulation Conference.

Howick, S. (2001). Using system dynamics models with ligitation audiences. Glasgow, Strathclyde Business School.

Howick, S. (2002). Should system dynamics be used to model distruption and ligitation? An exploration of criteria. Glasgow, Strathclyde Business School.

Howick, S., and C. Eden (2002a). Organizational learning and litigation analysis in project management: on the nature of discontinuities in system dynamics modeling of disrupted projects. Glasgow, Strathclyde Business School.

Kelton, W. D., R.W. Sadowski, and D.A. Sadowski (2001). Simulation with Arena. London, McGraw-Hill.

Kerzner, H. (1992). Project Management - A Systemic Approach to Planning, Scheduling and Controlling. New York, Van Nostrand Reinhold.

Kerzner, H. (2000). Applied Project Management: Best Practices on Implementation. New York, Wiley.

Lyneis, M. J., K.G. Cooper, and S.A. Els (2001). "Strategic management of complex projects: a case study using system dynamics." System Dynamics Review **17**(3): 237-260.

MacMaster, G. (2000). "Can we learn from project histories?" PM Network Magazine **14**: 66-67.

Morris, P., and G. Hough (1987). The Anatomy of Major Projects. Newyork, Chichester, Wiley.

Ondash, C. S., J. M. Huerta, and M. Stephen (1988). Large project simulation: A powerful tool for project management analysis. Winter Simulation Conference.

Ottjes, J. A., and H.P.M. Veeke (2000). Project management with simulation - A critical view on critical path. ICSC Symposia on Intelligent Systems & Applications, Wollongong, Australia.

PMI, P. M. I. (2000). Project Management Book of Knowledge (PMBOK). Upper Darby, PA, Project Management Institute.

Repenning, P. N. (1999). Resource dependence in product development improvement efforts. Cambridge, MA, Sloan School of Management, MIT.

Repenning, P. N. (1999b). A simulation-based approach to understanding the dynamics of innovation implementation. Cambridge, MA, Sloan School of Management, MIT.

Rodrigues, A. G. (1998). SYDPIM - A system dynamics-based project management integrated methodology. Glasgow, Strathclyde Business School.

Rodrigues, A. G., and T.M. Williams (1996a). System dynamics in software project management: towards the development of a formal integrated framework. Glasgow, Strathclyde Business School.

Rodrigues, A. G., and T.M. Williams (1996b). System dynamics in project management: assessing the impacts of client behaviour on project performance. Glasgow, Strathclyde Business School.

Rodrigues, A. G., J.A. Bowers (1996c). "System dynamics in Project management:a comparative analysis with traditional methods." System Dynamics Review **12**(2): 121-139.

Schlichter, J. (2001). PMI's organizational project management maturity model:emerging standards. PMI Annual Conference 2001, Upper Darby, PA,USA, PMI.

Sterman, J. D. (1984). "Appropriate summary statistics for evaluating the historical fit of system dynamists models." Dynamica **10**(II).

Sterman, J. D. (1992). System Dynamics Modeling for Project Management. Cambridge, MA, Sloan School of Management, MIT.

Sterman, J. D. (2000). Business Dynamics, System Thinking and Modeling for a Complex World. Boston, Irwin McGraw-Hill.

Turner, J. R., R.A. Cochrane (1993). "Goals-and-methods matrix:coping with projects with ill-defined goals and/or methods of achieving them." International Journal Of Project Management **11**: 93-102.

Vlatka, H., and S. Robinson (1998). Business process modeling and analysis using discrete-event simulation. Winter Simulation Conference.

Williams, T. (1999). "The need for new paradigms for complex projects." International Journal Of Project Management **17**(5): 269-274.

Williams, T. (2001). Assessing extension of time delays on major projects. Glasgow, Strathclyde Business School.

Williams, T. (2002). Learning from projects. Glasgow, Strathclyde Business School.

Williams, T. (2002a). Modeling Complex Projects. Warminster, John Wiley & Sons, Ltd.

Williams, T. (2003). Why Monte-Carlo Simulations of Project Networks are Wrong. Glasgow.

Williams, T., C. Eden, F. Ackermann, and S. Howick (2001). The use of project post-mortems. Glasgow, Strathclyde Business School.

Yourdon, E. (1979). Managing the Structured Techniques. Raleigh, Yourdan Press.

Appendix - Applications of System Dynamics to Project Management

Author	Years	Project type	Summary
Roberts	1964	R&D	Perceived vs. real progress
Kelly	1970	R&D	Development of R&D dynamics, multi-project management
Richardson, Pugh	1981	R&D	Productivity and rework generation staff hiring policy
Jessen	1988	R&D, construction	Project team motivation and productivity, client and project team relationship
Keloharju, Wolstenholme	1989	R&D	Time-cost trade-off
Abdel-Hamid	1988-1993	Software development	Project staffing policies, multi-projects scheduling, quality assurance policies, cost and schedule estimates as targets, managerial turnover
Barlas, Bayraktutar	1992	Software development	Simulation based game, staffing policies
Pugh-Roberts Associates	1993	Various large projects	PMMS: a specialist SD project management tool, design and workscope changes, dispute resolution
Smith et al.	1993	Software development	Charles Stark Drapper Laboratory
Chichakly	1993	Software development	High Performance System Inc, technology transition
Lin	1993	Software development	NASA Jet Propulsion Laboratory: integrating engineering and management
Aranda	1993	Software development	Aragon Associates Inc, TQM and product life cycle
Cooper, Mullen	1993	Software development	The rework cycle, project monitoring progress ramps
Williams et al.	1995	Product development	Dispute resolution, impact of parallelism
Rodrigues, A.G.; Williams, T.M	1996	Software development	System dynamics integrated to traditional project management (a hybrid approach). Implemented for BAeSEMA Company.
		Software development	System dynamics integrated to traditional project management (a hybrid approach). Implemented in KDCOM project for Korea Navy
Kimberly and James	1999	R&D	Discussing the rework cycle, feedback effects, and knock-on effects which create budget and schedule overrun