Project Title: Process Specification and Simulation

Investigators: Kincho H. Law and Shawn Kerrigan, Stanford University

Duration: January 1, 2000 – December 31, 2000

Project Objectives

The objective of the proposed study is to evaluate the process specification language (PSL) and a simulation query language (SimQL) with application to project and workflow management. This proposed joint research effort with NIST includes:

- to evaluate the potential of the Process Specification Language (PSL) for the planning and modeling activities in a project
- to evaluate the adequacy of the Process Specification Language (PSL) as an interchange definition language to support process-oriented simulation
- to evaluate the applicability of SimQL, a simulation query language, for practical engineering problems
- to develop an integration framework using PSL and SimQL for process-oriented simulation

Our long term goal is to develop a distributed network-based framework to integrate process specification and modeling and virtual simulations of project activities. To facilitate this research, we plan to use the research software VDT developed at Stanford's Center for Integrated Facility Engineering as a benchmarking application for the evaluation of PSL and SimQL. VDT is a project and organization modeling system designed to assist in developing organizational structures and identifying potential problems with project cost, time, or quality.

In the following, we briefly summarize PSL, SimQL and VDT. We then describe the extensions that may be needed in PSL for a VDT type project management tool. Finally, we discuss an integration framework utilizing PSL and SimQL.

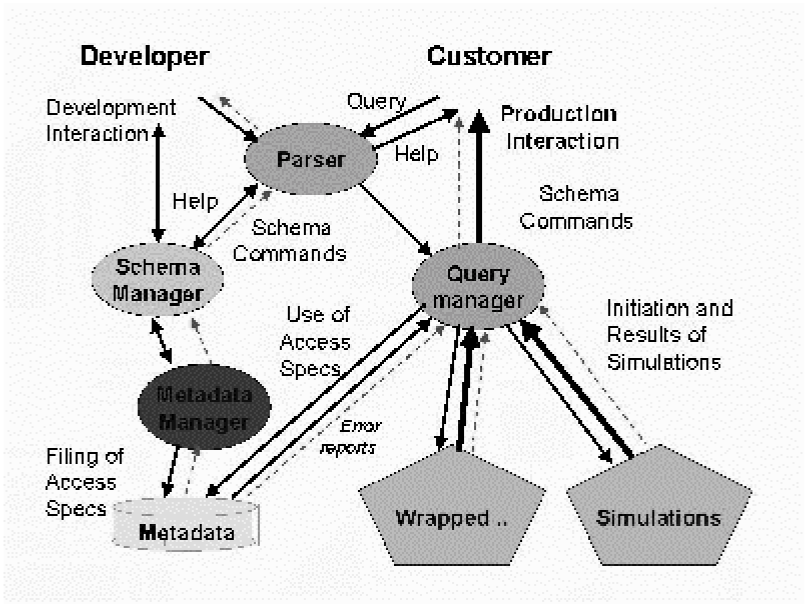
PSL

PSL is currently being designed at NIST as a process interchange language. The goal of the project is to make PSL generic enough to be independent of any specific application, but still robust enough to be able to represent process information for any given application. PSL is intended to include a default interchange language that manufacturing applications can use to communicate. Current efforts have focused on developing the PSL data model (also referred to as an ontology) which will define the meaning of the process-related terms in the language. In the current form, it is more appropriate to refer to PSL as an ontology or a data model written in KIF. Future plan at NIST includes incorporating a syntax and grammar specification to make it a language. It is noted that the motivation of PSL is not to develop a process modeling language (i.e., a language to serve as an underlying representation for manufacturing applications), though there is nothing in the work that prohibits PSL from serving this function in the future.

SimQL

The concept of process-oriented simulation languages has existed for quite some time, particularly for simulating the performance of computer systems. Such languages are typically employed to write simulation tools for single application. SimQL, on the other

hand, is not designed as a language for writing simulations, but instead as a language for providing a common wrapper and standardized data model for interaction with simulation programs. The motivation behind SimQL is that the simulation should be able to be written in any language using any technique, but accessed through a consistent language independent wrapper. This situation is analogous to SQL, where the database is written in C or some other language, but SQL provides a uniform method for accessing the data. SimQL is a simulation access language that has the potential to standardize and improve the reusability of simulation software.
SimQL consists of a model schema which provides the user with available variables and parameters, and a query language for invoking the simulation with the parameters described in the model schema. The query schema is largely intended to be built into an application by programmers, rather than be utilized directly by the user. The query calls the simulation with all the parameters defined by the model schema. The results of the query invocation return the values specified in the model schema. The standard schema and query model allows application programmers to easily incorporate existing simulation programs into new simulation applications. SimQL is a powerful language for wrapping existing simulations, and as such has many potential applications. The overall layout of SimQL system components is shown below (Ref: Gio Wiederhold, "A Simulation Access Language (SimQL)", http://www-db.stanford.edu/LIC/SimQL.html)



VDT
VDT takes traditionally qualitative organizational management theory, and builds a model that incorporates some rough quantitative measures. It does this with a hierarchy of "actors" who manage specific activities and report problems up the chain of command to superiors. An actor can be an individual, a part-time worker, or a team. The activities are managed by processing the activity duration. Actors send and receive messages regarding "Requests for Information" and rework instructions.
VDT builds upon the ideas of CPM by adding such notions as rework, failure dependency, and communications/coordination information to the activity diagram. The

user can specify the rate of activity subtask errors. If an error is found in an activity subtask, rework may be required for the given activity, along with activities that have failure dependencies with the activity in error. The decisions regarding rework depends on the user's input (probabilities associated with rework) and the management structure. The rework decisions will affect the project's time, cost, and quality factors in the simulation. In essence, VDT links together a management/organization model and an activity model. The tool allows users to change information about management or activities, and view the effect on the total project.

A VDT project is composed of a traditional CPM diagram, additional links showing failure dependence and reciprocal information dependence, a management structure diagram, and responsibility links between the management structure and activities. The following example project of a microprocessor design from the VDT software tutorial is used to clarify these components.

1. First we will develop a CPM activity diagram as shown in Figure 1. Activities are named and given durations, and groups of activities are linked together using relationships such as finish-start, start-start, or finish-finish. Milestones may be included in the activity diagram, such as "ship tapes to foundry" in the example project. Begin and end milestones are used to denote the start and finish of the project ("Start Project" and "Fab, Test and Deliver.")
2. Failure dependence and reciprocal information dependence can be added to the CPM diagram as shown in Figure 2. A failure dependence link indicates that failure of a subtask in an activity may result in required rework for a previously completed subtask in the failure dependent activity. These links are added to the CPM diagram as (red) dashed lines. Reciprocal information links indicate that the actors responsible for the activities will send communications to each other at the end of subtasks making up the activities. Reciprocal information links are added to the CPM diagram as (green) dashed lines.
3. Adding actors, such as a chip architect or foundry test engineer, to the diagram creates a management structure diagram. The actors are linked together by "reports-to" relationships. A sample management structure for the chip design team is shown in figure 3.
4. Actors are responsible for particular activities. Figure 4 shows the "responsible-for" links that bring the management structure and the activity chart together. The contribution by the actor to the activity is listed on the "responsible-for" link.
5. Meetings are scheduled as recurring activities that help communication and coordination but also consume actors' time. Figure 5 shows the complete VDT model diagram where the managers (project lead, chip architect, and foundry lead) are required to attend meetings.

Preliminary Evaluation of PSL for VDT Application

Currently PSL is focused on modeling activities' duration and ordering relationships. In its present form, PSL is currently insufficient for modeling the more complex ideas introduced in VDT, such as uncertainty, failure dependency, actors, and actor skill sets. This does not imply that PSL is not capable of representing these ideas. The power of PSL lies in its flexibility and potential for growth and expandability. With extensions, PSL can be expanded to incorporate the concepts and ideas present in the VDT model.

We categorize five extensions that may be needed for PSL to incorporate the information used by VDT. These extensions would add the capability to model activity failure and communication dependence, management structure, activity attributes, actor attributes, and project attributes. The first two extensions described below, the activity failure and communication dependence extension and the management structure extension, would be useful standard extensions to the PSL language. The other extensions would be required to fully implement VDT in PSL, but are rather VDT specific and may therefore be primarily useful for further testing of PSL's flexibility in terms of expansion capabilities. The activity failure and communication dependence extension would add the following concepts to PSL:

- Failure-dependence – relationship would add the capability to indicate when failure in an indicated successor activity might result in rework for the specified activity
- Reciprocal-information – relationship indicates when communication is required between the managers of the related activities.

1. The idea of actors, or activity managers, is not currently present in PSL. This idea could be incorporated in a management structure extension, which would extend the idea of objects from PSL – core. The new definitions in this extension would include:

- Actor – object to represent a person or group of people working as a team
- Reports-to – defines a hierarchy of actors according to who they report to.
- Responsible-for – relationship between actors and activities defining what activities an actor is responsible for.

1. The activity attributes that cannot currently be represented in PSL but could be encapsulated in an activity attribute extension are:

- Description – activity description
- Requirement complexity – (low, medium, high) complexity rating that depends on the number of internal requirements this activity must satisfy.
- Required skill – skill required to complete the activity quickly
- Solution complexity – (low, medium, high) complexity rating that depends on the number of solutions (activities) to which the activity contributes.

Similar to the activity attribute extension, an actor attribute extension would be required for VDT in PSL. The required actor extensions are:

- Actor-description – description of the actor
- Actor-size – number of people making up the actor, in terms of full-time equivalent workers
- Actor-skill – skills possessed by the actor
- Actor-skill-level – (low, medium, high) skill level for a particular skill possessed by the actor
- Cost – cost per unit time for the actor's work
- Task-experience – (low, medium, high) the actor's experience for this type of project
- Organization-role – the actor's role in the organization (sub-team, sub-team leader, project manager)

A project attribute extension to incorporate ideas for VDT would cover general project information such as:

- Project-description – description of the project

- Work-day – (hr) number of working hours per day (i.e. typically 8)
- Work-week – (days) number of working days per week (i.e. typically 5)
- Decision-centralization – (low, medium, high) characterization of centralization of decision-making responsibility within the project.
- Communication-formalization – (low, medium, high) characterization of formalization of communications in memos or organized meetings.
- Team-experience – (low, medium, high) previous experience of the project team working together
- Activity-failure-probability – (percentage) probability that each subtask comprising the activity fails the verification that occurs as it is completed.
- Activity-internal-rework – (percentage) probability that a sub-task failure results in internal rework for the activity
- Activity-dependent-rework – (percentage) probability that a sub-task failure results in rework for failure dependent activities
- Communication-noise – (percentage) probability that communication is non-activity related "noise"

Summary and Discussion

VDT presents an excellent test bed for the use of PSL as a translation and integration process specification language. VDT requires information from what are generally considered two very disjointed areas: management structure and CPM activity charts. There are multitudes of CPM scheduling programs available, and any one of them could be used to generate the CPM charts for translation into PSL. An organizational structure chart could be generated in another program and translated into PSL. From the pool of PSL data, a translator could generate a nearly complete VDT project simulation. With minimal input into the VDT program, users could identify potential trouble areas in the project and tinker with management structure and activity schedules to improve the effectiveness of the overall project design. A longer-range goal could be to enable the active communication of these programs. This would mean changes in VDT would automatically register within the organizational design program or the CPM scheduling program.

The application of SimQL in conjunction with PSL to a program such as VDT could be very useful. VDT is a project and organization modeling system that generates results via multiple simulations. Numerous simulations are run, averaged, and presented to the user as predicted project results. In essence, VDT is a program that could be wrapped using SimQL, provided all the necessary input data (and there is a lot of required data) could be gathered in advance. A process specification language, such as PSL, could be used to gather the required input data from various CPM and management organization programs. The pool of data could be used to query VDT through a SimQL wrapper. The results of the query would supply all of the VDT output for the simulation. This virtual environment would increase the value of simulations and aid in the design of integrated total project lifecycle applications.

The power of a standard query method for simulations could be very powerful in a CORBA type environment. In our current work, we have been experimenting with a distributed internet-CAD framework. The three layer model includes a communication layer, a standard product model layer and the core application. In the process-oriented framework, we envision a similar architecture that also includes a process specification standard and a simulation query facility. The process specification will provide a guide to the tasks to be simulated and the simulation query will act to inquire and access the applications. PSL is a powerful new language for process specification. Further development and expansion of this language is required for PSL to reach its full potential. VDT provides a valuable benchmark for the incorporation of new ideas in PSL, as well as an interesting platform for the integration of PSL data gathered from multiple programs. SimQL could provide the link to facilitate access to multiple simulation programs.

Proposed Tasks

This proposed project is intended to be a feasibility study for the future development of a distributed workflow platform utilizing PSL and SimQL. Standard specification language to describe process models and standard query language to access simulation packages are fundamental to such distributed applications. The project will include a thorough evaluation of the current PSL and SimQL. In particular, a specific project management tool, VDT, will be employed throughout this investigation. A simple project description using the chip manufacturing example will be used to validate the applicability of PSL and SimQL.

The proposed research tasks for the one-year period are:

- to evaluate the adequacy of PSL and to propose extensions for project management application software (3-6 months);
- to evaluate the adequacy of SimQL for the proposed application problem (3-6 months);
- to develop an integration framework for process oriented simulation (3 months).

Based on the results of this feasibility study, we anticipate that extensions to PSL and SimQL will be proposed and developed. During this proposed feasibility phase of study, we plan to examine the issues related to the application of PSL and SimQL to workflow management applications. An investigation plan for future tasks will be developed during the current proposed study period.