Effective Resource Allocation for Process Simulation: A Position Paper

Mohammad S. Raunak and Leon J. Osterweil
University of Massachusetts at Amherst
{raunak, ljo}@cs.umass.edu

***Abstract*—One of the primary reasons to execute or simulate processes is to be able to reason about, forecast, and plan the best utilization of available resources. As process programmers, we define resources to be the agents that carry out tasks, and the tools and other entities required by agents in order for them to be able to complete their assigned work. Specifying these resources rigorously and allocating them efficiently during process simulation or execution is a non trivial problem. In this paper, we present many hard and interesting issues related to resource management and propose some solution approaches. In particular, we talk about an auction based solution approach, which we feel fits well in different types of process simulation.**

***Index Terms*— allocation, process, resource, simulation**

I. INTRODUCTION

One of the primary reasons to execute or simulate processes (and many other types of software) is to be able to reason about, forecast, and plan, the best utilization of available resources [kellner99]. Managing resources well translates into better process management, which in turn, ensures better quality in the final products or services resulting from a process. From our perspective as process programmers, we define resources to be the agents that carry out tasks, and the tools and other entities required by agents in order for them to be able to complete their assigned work. Specifying these resources rigorously and allocating them efficiently during process simulation or execution is a non trivial problem.

Our previous work in process programming has focused on specifying the coordination of activities, their execution semantics and artifact flows [wise98, cass99, cass00]. All these issues are undeniably crucial for properly capturing a process and supervising its execution. However, we have always maintained that a clear, precise and complete definition of the resources required by the process, and flexible mechanisms for their allocation, are no less important for process analysis and improvement. We argued the importance of a separate resource manager and laid out a modeling approach in earlier works [rodion99, lerner00]. Over the last few years our experience with process modeling, analysis, and execution have grown, drawing from our work with processes in such diverse domains as digital government, e-commerce, and medical services, as well as with software process. These experiences have provided insights into some more challenging issues and have pointed us toward some promising approaches to address the challenges. In this paper, we present many hard and interesting issues related to resource management and propose some solution approaches.

This research was partially supported by the Air Force Research Laboratory/IFTD and the Defense Advanced Research Projects Agency under Contract F30602-97-2-0032, by the U.S. Department of Defense/Army and the Defense Advance Research Projects Agency under Contract DAAH01-00-C-R231, and by the National Science Foundation under Grant No. CCR-0204321. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied of the Defense Advanced Research Projects Agency, the Air Force Research Laboratory/IFTD, the U.S. Dept. of Defense, the U. S. Army, The National Science Foundation, or the U.S. Government
.

II. RESOURCE MANAGEMENT CHALLENGES

In our work we define a process as (largely) hierarchical structure of tasks, called steps. Each step is to be executed by an agent, whose characteristics are specified as part of the process definition, and a set of auxiliary resources, also specified as part of the process definition [cass00, wise98]. In our approach, the execution of such a process definition entails the binding of a specific agent, and specific resources, identified by means of a search of a resource repository, to each process step, as it comes up for execution. An important premise of our work is that resources, and indeed agents, can be either humans or automated devices (either hardware or software). Thus our process definitions can rightly be viewed as specifications of how humans and automated devices are to be coordinated. This view particularly emphasizes the importance of effective identification of the agents and resources to be selected for binding to the various steps. Invariably the resources that are available are in short supply and are the subjects of contention. Thus, potential delays, bottlenecks, and resource starvation can presumably be avoided or reduced by using resource analysis to identify ways in which resource contention can be alleviated (eg. by causing tasks to execute in parallel, or by skillful sharing of resources).

Another way to achieve better process performance is to schedule the resources based on some allocation strategies. Researchers in operating systems, multi agent systems and operations research have long looked at different allocation algorithms for better utilization of resources in processes. However, the challenges we face in software and a lot of other processes are unique due to the diverse nature of resources we need to deal with here. Resources like processor time, network bandwidth or completely automated agents have strictly predictable behavior that is relatively less complex to schedule and reason about. The simultaneous presence of humans and automated agents, and the big difference in the time granularity of the required resources, make it inherently much more difficult to specify them properly and allocate them effectively with a coherent scheduling scheme.

There are three very important and intricately interconnected tasks in the realm of resource management: modeling the resources, identifying and retrieving the resources and finally, scheduling the resources. Usually, the modeling task usually takes place before simulation, potentially in parallel with the process definition work. The modeling or specification of resources, to a large extent, is dependent on the domain for which a process is being programmed. In software processes, for example, we have human agents like programmers, designers, testers as well as non-human resources compilers, development environments, software licenses etc. On the other hand, a medical emergency room process includes resources like, nurses, doctors, orderlies, beds, medicines, and a whole lot of other equipments. It is quite challenging to provide a common structure that would allow a process programmer to model resources from these diverse domains coherently.

Identifying and retrieving the resources require capabilities to store resource information persistently and to be able to query them with some rich query language. Specifying and storing both type definitions and resource instances adequately is always challenging. One has to be careful in defining the Resource Manager architecture to ensure flexibility in supporting processes from diverse domains with a wide ranging resource types.

Scheduling of resources is primarily required during run time. The modeling and scheduling capabilities that we have previously proposed in [rodion99] have proven effective in programming processes with comparatively less complex resource variety and requirements. Our previous approach worked by statically defining the resource model and letting process steps query that static model. However, we have discovered that agents and resources often have dynamic attributes which may change in real time during the execution of the process. For example, in an emergency department process of a hospital, the nurses (one type of agent/resource) are frequently changing locations and any query based on the propinquity of a resource requires the *resource manager* to allow for a very dynamic definition of resources.

Software, medical, and other processes require complex specification of composite resources like design teams or nursing pools which include instances of individual agents, each of which might potentially need to be allocated to several different tasks. While allocation of such resources can be done by modeling capacity of these resources and to allow them to be assigned to multiple tasks in parallel, a more accurate model of such partial allocation would require time based allocation of the agents to each task. In this case, an agent will not be bound to a process step indefinitely; it will be acquired by the step for a specific period of time.

Often we come across process programs where resources are sought in some preferential order. For example, there may be a designated agent for a task; however, other agents with some particular skill can carry out the task in absence of the designated agent. The process definition needs to specify the preferred resource in addition to specifying all the attributes properly to identify a resource properly.

Our experiences with processes and resources have shown that there are process programs that need to define situations where the simulation or execution of the process should block in case of a resource being busy. Allowing such blocking calls to the resource server during an execution or simulation of a process gives rise to the whole dimension of potential deadlocks and starvations.

An important requirement of resource management is the ability to reserve resources for planning purposes. This service requires look-ahead capabilities on the process execution paths. As process programmers, we often need to deal with complex processes with abundant exceptions. Such control flow structures make it very difficult to predict future resource requirements based on which proper reservation can be made. A related requirement for a resource manager is to be able to identify the level of resource redundancy required to ensure safe completion of processes in case of a sudden surge in activities.

Our ongoing work with medical processes has strongly demonstrated the need for resource preemption capability in the resource manager. Resources like doctors, nurses or even hospital beds get preempted and assigned to a new instance of a task with higher priority in an emergency department process. We are convinced that such cases of higher priority work preempting resources from a running low priority task is not uncommon in software processes either. From the process simulation or execution perspective, this brings in a whole new level of complexity.

III. Solution Approaches

In [lerner00], we described a resource model based on *resource classes*, *resource instances*, their *attributes* and *relationships* amongst them. Our subsequent experience with process simulation work has shown that not all the elements of that proposed framework are necessarily effective for resource modeling and allocation in simulations. We thus propose a new approach to resource modeling where entities are primarily categorized as agents and non-agent resources. Our contention is that the association between a process step and its required resources is driven by constraints on the agent or agents responsible for carrying out the task.

Resources usually have different types of constraints. Amongst others, there are “requires” relationships between resource entities which prevent assigning of a resource without also assigning one or more other required resource. This constraint can be addressed by having attributes associated with the resources that would allow the resource manager to query on a resource attribute to create compound resource collections dynamically at run time.

The other important constraint on resources is brought about by dynamically created composite objects. For example, a programmer pair in XP pair programming practices or a couple of nurses assigned to a trauma patient. We propose to utilize ideas similar to “views” used by database researchers to create abstract tables on the fly. The process programmer will define resource objects and will provide definitions of the types of composition allowed for these resource objects. When a process step specifies a query for one of these composite resource objects, a query processor will join multiple objects to create composite resource on the fly and the Resource Manager will retrieve them atomically.

We also propose the incorporation of timing constructs in our process model, Little-JIL [wise98], in order to provide important information regarding the time within which an agent needs to start its work after being assigned to a task, and the maximum time the agent is given to complete the task. In some cases, we also envision the need for specifying the minimum time required for a task for scheduling purposes.

The resource analysis study is often performed with the objective of identifying the amount of redundancy required for resources to successfully execute processes in case of a surge of activity. We are in the process of collecting real life data for a medical emergency department process to investigate the characteristics of process behavior and resource loads under such circumstances. We also plan to explore process simulation with stochastic input model to create such scenarios.

We will explore the allocation of resources using strategies based on known scheduling algorithms from other areas like operating systems for non-agent resources. However, we propose a novel approach for the assignment of agents to tasks. Agents and groups of agents will be presented with specifications of tasks, time requirements and other resource requirements related to them. Assignment of agents to tasks is to be done by means of an auction. The agents will be required to bid for assignment to these tasks, being incentivized to bid as aggressively as possible, consistent with private valuations based upon their need for work, desire for advancement, or fear of layoffs. The final assignment of tasks will be done in response to determination of the highest bidder. Groups of agents in this scenario will be allowed to communicate amongst themselves in a manner that is similar to the way bidders may collude in other types of auctions. In our case, however, bidder collusion will be aimed at coming up with the highest bid for a task.

We believe this idea of auctioning is going to be applicable in multiple areas of different resource management issues. The preferential ordering of resources that we have discussed earlier can be achieved based on the bids placed by the different agents (and other resources). We also feel that the requirements of supporting dynamic creation of composite resource objects can be aptly addressed with the utilization of combinatorial auctions. There are a lot of literature in Operations Research on different auction formats and their effectiveness. We believe, we can utilize those results and experiment with them in our environment.

IV. Related Works

Different software process programming languages like APEL[establier97], MVP-L[rombach93], ALF[canals94], Statemate[harel90] and Process Weaver[fern93] have used resource managers to facilitate process execution. However, the modeling capabilities in such systems are restrictive and do not provide support for scheduling.

There has been considerable work in operating systems focusing on scheduling techniques of primarily hardware resources [goyal96, shenoy98]. As mentioned earlier, the domains of their work including required timing granularity, differ significantly from resources that include both human and automated agents as well as other entities.

V. Concluding remarks

Modeling as well as scheduling of resources to reason about effective process execution and resource requirements is a hard problem. There is ample literature in other areas of computer and management sciences that look at the resource management issues from different domain perspectives. It is crucial for our community to take a critical look at the issues of resource management, identify the unique challenges faced by the process programmers while simulating or executing processes, determine what approaches from other areas may or may not work, and propose and evaluate new solution approaches. In this paper, we have tried to motivate such resource management research by pointing to some intricate issues related to resources within a process environment. We have also proposed an auction based solution approach, which we feel will work well in different types of process simulation and execution.

Acknowledgment

We would like to thank Barbara Lerner, Rodion Podorozhny, Anoop Ninan and Joel Sieh for their earlier attempts at developing some initial versions of resource management service for process execution support.

References

[canals94] Canals, G. et. al., ALF: A framework for building process-centered software engineering environments. In Software Process Modeling and Technology, pages 153-185Researhc Studies Press, Sommerset, England 1994

[cass99] Cass, A.G., Lerner, B. S., McCall, E. K., Osterweil, L. J., Sutton, Jr., S. M. and Wise, A. Logically central, physically distributed control in a process runtime environment, *Technical Report 99-65, University of Massachusetts*, Department of Computer Science, November 1999

[cass00] Cass, A.G., Lerner, B. S., McCall, E. K., Osterweil, L. J., Sutton, Jr., S. M. and Wise, A. Little-JIL/Juliette: A process definition language and interpreter, In *Proceedings of the 22nd International Conference on Software Engineering*, 754-757, Limerick, Ireland, June 2000

[establier97] Establier, J., Dami, S. and Amiour. A., APEL: A graphical yet executable formalism for process modeling, *In proceedings of Automated Software Engineering*, March 1997.

[fern93] Fernstrom, C. PROCESS WEAVER: Adding process support to UNIX. *In the second International conference on Software processes*, pages 12-26, 1993 `

[harel90] Harel, D. and Lachover, H. et. al. STATEMATE: A working environment for the development of complex reactive systems. *IEEE Transactions on Software Engineering*, vol 16, issue 4, April 1990.

[kellner99] Kellner, M. I., Madachy, R. J., and Raffo, D. M., Software Process Simulation Modeling: Why? What? How? *Journal of Systems and Software*, vol. 46, no. 2/3, 15 April, 1999

[lerner00] Lerner, B. S., Ninan, A. G, Osterweil L. J., Podorozhny R. M. , Modeling and Managing Resource Utilization in Process, Workflow, and Activity Coordination, Technical Report, Department of Computer Science, University of Massachusetts, Amherst, MA 01003, August 2000. *(UM-CS-2000-058)*

[rodion99] Podorozhny, Rodion, Lerner, B. S. and Osterweil L. J., Modeling Resources for Activity Coordination and Scheduling. *3rd Internation conference on coordination models and languages*, Amsterdam 1999

[rombach93] Rombach, H. and Verlage, M. How to assess a software process modeling formalism from a project member's point of view. *In proceedings of the Second International Conference on Software Processes,* pages 147-159, 1993.

[wise98] Wise, Alexander, *Little-JIL 1.0 Language Report*, Department of Computer Science, University of Massachusetts, Amherst, MA 01003, April 1998. *(UM-CS-1998-024)*