Towards an Integrated Planning and Adaptive Resource Management Architecture for Distributed Real-time Embedded Systems

Nishanth Shankaran[†], John S. Kinnebrew[†], Xenofon D. Koutsoukos[†],
Chenyang Lu[‡], Douglas C. Schmidt[†], and Gautam Biswas[†]
[†]Dept. of EECS Vanderbilt University, Nashville, TN
[‡]Dept. of Computer Science and Engineering, Washington University, St. Louis, MO

Abstract

Distributed real-time embedded (DRE) systems often operate in open environments where operating conditions, input workload, and resource availability cannot be accurately characterized a priori. Some DRE systems, such as NASA's Magnetospheric Multi-Scale (MMS) mission, perform sequences of heterogeneous data collection, manipulation, and coordination tasks to meet specified objectives/goals. These systems are also required to operate with a high degree of local autonomy and adaptivity as new data is acquired and analyzed, and as environmental conditions change. Key challenges in managing open DRE systems include effective planning and online management of system resources to accommodate for changing mission goals, environmental conditions, resource needs, and resource availability. This paper explores the benefits of an integrated planning and adaptive resource management architecture that combines decision-theoretic planning with adaptive resource management to control and ensure efficient functioning of open DRE systems.

1 Introduction

Emerging trends and challenges. *Distributed real-time and embedded* (DRE) systems form the core of many mission-critical applications that operate in dynamic and uncertain environments. An example of such an application is NASA's Magnetospheric Multi-Scale (MMS) mission [1]. As with many other DRE systems, the MMS mission system operates in *open* environments where operating conditions, input workload, and resource availability cannot be accurately characterized *a priori*. Moreover, overall mission goals and analysis methods may change as new data and information is obtained or scientists that operate the mission propose new goals/objectives.

Conventional resource management approaches, such as end-to-end task allocation and scheduling [2], are designed to manage system resources and maintain QoS in *closed* environments where the operating conditions, input workloads, and resource availability are known in advance and do not change much at runtime. These solutions, however, are insufficient for complex DRE systems such as the MMS mission system that operate in open environments since the autonomous operation of these systems require them to *adapt* to a combination of (1) changes in mission requirements and goals, (2) changes in operating conditions, (3) loss of resources, and (4) drifts and fluctuations in system resource utilization and application QoS at runtime.

Adaptation in such complex DRE systems can be performed at various levels, including at the (1) *system level* by deploying/removing end-to-end applications to/from the system, (2) *application structure level* by adding, modifying, and removing components or sets of components associated with one or more applications executing in the system, (3) *resource level* by reassigning resources to application components to ensure their timely completion, and (4) *application parameter level* by fine-tuning configurable parameters (if any) of application components. These adaptation decisions, however, are tightly coupled as they directly or indirectly impact the utilization of system resources and QoS of the system, which ultimately defines the success of the overall mission. It is therefore necessary that adaptations at various levels of the system are performed in a stable and *coordinated* fashion.

Solution approach → Integrated planning and adaptive resource management. To address the planning and adaptive resource management needs of open DRE systems, we developed the *Spreading Activation Partial Order Planner* (SA-POP) [3] and the *Resource Allocation and Control Engine* (RACE) [4], respectively. SA-POP combines decision-theoretic task planning with resource constraints for DRE systems operating in uncertain environments to produce effective executable component sequences that can achieve current mission goals, including desired QoS, with available system resources. RACE provides a customizable and configurable adaptive resource management framework that enables open DRE systems to adapt to fluctuations in utilization of system resources and QoS specifications.

Although SA-POP provides efficient re-planning capabilities in response to changes in the goals of applications, it cannot deal with uncertainties and fluctuations in task execution times and resource availability. To enhance the robustness of the system, RACE support adaptive resource management at two levels: (1) task reallocation in response to significant changes in component sequences caused by application re-planning, and (2) feedback-control-based

task rate adaptation. Note that the two adaptation strategies are complementary to each other. Task reallocation provides coarse-grained adaptation to application changes, while task rate adaptation provides fine-grained adaptation to resource fluctuations.

Our experience developing a MMS mission prototype [5] showed that although SA-POP and RACE performs effective adaptive resource management, neither SA-POP nor RACE, individually, have sufficient capabilities to efficiently manage and ensure proper functioning of such complex DRE system. To meet the challenge of such open DRE systems, we propose an integrated planning and adaptation resource management architecture.

2 An Integrated Planning and Resource Management Architecture

This section first describes SA-POP, RACE, and our integrated planning and adaptive resource management architecture. It then shows how we applied this integrated architecture to address the QoS needs of our MMS system prototype.

2.1 Overview of SA-POP

Open DRE systems can operate more efficiently and effectively by incorporating some degree of autonomy that allows them to adapt to changing mission goals and environmental conditions. SA-POP provides a planning approach for dynamic generation of component-based applications that operate with limited resources in uncertain environments. The architecture of SA-POP is shown in Figure 1.

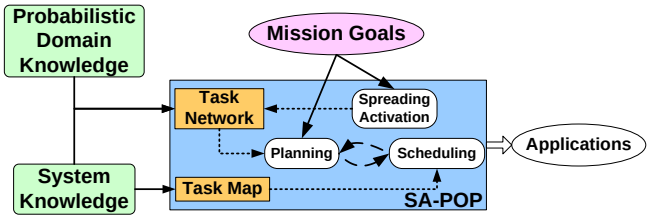

Figure 1. SA-POP Architecture

Given one or more goals specified by a software agent or system user, SA-POP takes into account current conditions to generate partial order plans with high expected utility [3]. Goals are specified as desired conditions with associated utility values. SA-POP uses a spreading activation mechanism [6] to generate expected utility values for individual tasks that contribute to achieving the specified goal conditions. Guided by these expected utility values, SA-POP's planning algorithm generates a set of task sequences that together achieve the goal while meeting all resource and time constraints.

For SA-POP to choose appropriate tasks to achieve a goal, it must know which preconditions must be satisfied for each task, its input/output data streams (if any), and the pertinent effects that result from its operation. Uncertainty as to whether tasks will produce the desired output or effects is captured via conditional probabilities associated with the preconditions and effects of a task. Together, these input/output definitions, preconditions/effects, and related conditional probabilities define the *functional signature* of the task.

To ensure applications and their scheduled executions do not violate resource and time constraints, SA-POP also requires knowledge of the expected resource consumption and execution time for each component/configuration that can implement a task, *i.e.*, its *resource signature*. The planning of SA-POP uses the task function signatures and associated component resource signatures to dynamically generate applications most suited to local conditions and resource availability. It also provides a schedule of acceptable time windows for the execution of each application component with any required before-after constraints on the execution components both within and between applications. Moreover, through re-planning, SA-POP can dynamically adapt deployed applications when environmental conditions and resource usage change unexpectedly.

2.2 Overview of RACE

RACE addresses two key challenges of adaptive resource management of open DRE systems: (1) efficient online resource allocation for applications, and (2) effective system adaptation in response to fluctuations in input workload, operating conditions, and resource availability. The RACE framework decouples adaptive resource management algorithms from the middleware implementation, thereby enabling the use of customized resource management algorithms without redeveloping significant portions of the middleware or applications. To enable the seamless integration of resource allocation and control algorithms into DRE systems, RACE configures and deploys feedback control loops.

Online resource allocation using RACE. As shown in

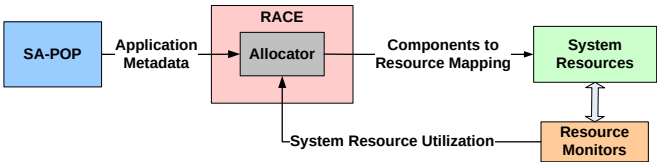

Figure 2. RACE: Online Resource Allocation

Figure 2, RACE features `Allocators` that implement resource allocation algorithms, such as multi-dimensional bin-packing algorithms [2], to allocate various domain resources (such as CPU, memory, and network bandwidth) to application components by determining the mapping of components onto nodes in the system domain. `Allocators` determine the component-to-node mapping at runtime based on *estimated* resource requirements of the components and current resource availability on the various nodes in the domain. As shown in Figure 2, input to

`Allocators` include the metadata of the application corresponding to the application generated by SA-POP and the current utilization of system resources.

Effective system adaptation. As shown in Figure 3,

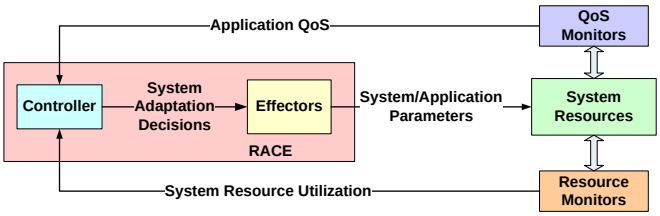

Figure 3. RACE: Online System Adaptation

RACE uses `Controllers` to implement control-theoretic adaptive resource management algorithms, such as EUCON [7], that enable DRE systems to adapt to changing operational context and variations in resource availability and/or demand. `Controllers` use the control algorithm they implement to compute system adaptation decisions to ensure that system performance and resource utilization requirements are met. Figure 3 also shows how these decisions serve as inputs to *Effectors* that modify system parameters (such as resources allocated to components, execution rates of applications, and OS/middleware/network QoS setting for components) to achieve the `Controller` recommended adaptation. RACE's `Controller` and `Effectors` work with resource monitors and QoS monitors to compensate for drifts/fluctuations in utilization of system resources and/or application QoS.

2.3 Overview of the Integrated Architecture

Our integrated planning and adaptive resource management architecture integrates SA-POP and RACE to address system management challenges of complex open DRE systems, such as the MMS mission system. Figure 4 shows the integrated SA-POP/RACE planning and adaptive resource management architecture. A set of QoS and resource monitors track system behavior and performance, and periodically update SA-POP and RACE with current resource utilization (*e.g.*, processor/memory utilization and power) and QoS values (*e.g.*, end-to-end latency and throughput). Although inputs to SA-POP and RACE include system behavior and performance metrics, SA-POP uses this information to monitor the evolution of the system with respect to its *long-term* plan/schedule for achieving given goals and to re-plan/re-schedule when necessary, whereas RACE uses this information to *fine-tune* application/system parameters in response to drifts/fluctuations in utilization of system resources and/or application QoS.

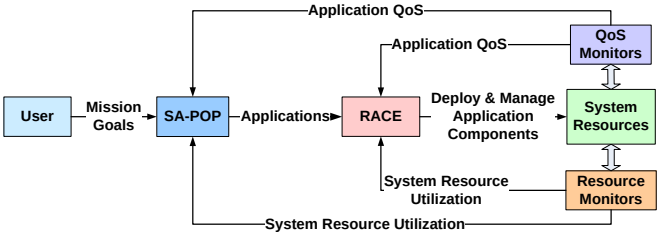

Figure 4. Integrated Architecture

Figure 4 also shows how the integrated SA-POP/RACE architecture is comprised of two hierarchical feedback loops: (1) the *inner feedback control loop* with resource and QoS monitors, RACE, and the DRE system, and (2) the *outer feedback control loop* that includes SA-POP, RACE, and the DRE system. The outer feedback loop enables a DRE system to adapt to new mission goals, major changes in resource availability (*e.g.,* loss of a satellite sensor or drastic deviations in resource consumption), and changes in environmental conditions, by performing *coarse-grained* system adaptation, such as adding/removing components deployed as part of an application and modifying the schedule for operation of components. The inner feedback loop computes *fine-grained* system adaptation decisions, such as fine-tuning application parameters (*e.g.*, execution rates) and system parameters (operating system and/or middleware QoS parameters), thereby compensating for drifts/fluctuations in utilization of system resources and/or application QoS.

2.4 Applying our Integrated Architecture to the MMS Mission System

As shown in Figure 5, our integrated planning and adaptive resource management architecture performs the following actions for the MMS mission system prototype:

1. Upon receiving a mission goal from the user, SA-POP employs integrated decision-theoretic planning to generate an application capable of achieving the provided goal, given current local conditions and resource availability.

2. After an appropriate application has been generated by SA-POP, RACE's `Allocator` allocates resources to application components and employs the underlying middleware to deploy and initialize the application.

3. RACE's `Controllers` and `Effectors` periodically compute system adaptation decisions and modify system parameters, respectively, to handle minor variations in system resource utilization and performance due to fluctuations in resource availability, input workload, and operational conditions.

4. RACE triggers re-planning by SA-POP if RACE's `Controllers` and `Effectors` are unable to adapt the system effectively to changes in resource availability, input workload, and operational conditions, *e.g.*, due to drastic changes in system's operating conditions, such as complete loss of resources. SA-POP performs iterative plan repairto modify the current application to achieve its goal even when it encounters unexpected conditions or resource availability.

Our integrated SA-POP/RACE architecture offers capabilities that: (1) efficiently handle uncertainty in plan-

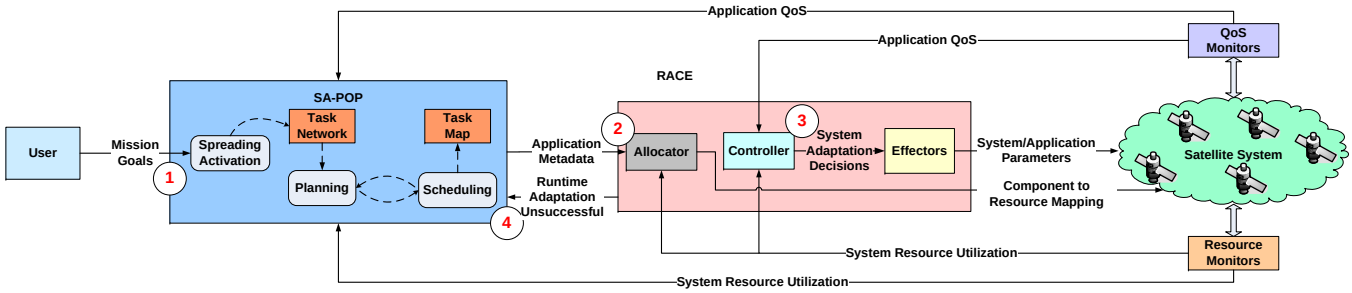

Figure 5. Applying the Integrated Architecture to the MMS Mission System

ning for online generation of applications, (2) planning with multiple interacting goals, (3) efficiently allocate system resources to application components, (4) avoid over-utilization of system resources, thereby ensuring system stability and application QoS requirements are met, even under high load conditions.

3 Concluding Remarks

The paper described how our integrated planning and adaptive resource management architecture helps address system management challenges of complex open DRE systems. By integrating SA-POP and RACE, our architecture enables open DRE systems to adapt to (1) high-level changes, such as new mission goals and drastic changes in operating conditions and (2) fluctuations in utilization of system resources and/or application QoS. Although SA-POP and RACE individually offer many features that enable the effective management of complex DRE systems, the benefits of our *integrated* architecture include:

Fast and efficient online planning and resource allocation. In our integrated architecture, SA-POP considers *coarse-grained* (system-wide) resource constraints during planning of applications. In contrast, RACE handles resource allocation optimization with *fine-grained* (individual processing node) resource constraints. The separation of concerns between SA-POP and RACE limits the search spaces in each during planning and resource allocation, thereby making the process of planning and online resource allocation faster, more effective, and more efficient.

Fast and efficient runtime system adaptation. Since planners such as SA-POP are designed to be domain independent, adaptation decisions computed by the planner are coarse-grained, such as adding/removing components deployed as part of an application and modifying the schedule for operation of some components. Moreover, system adaptation by SA-POP may involve significant computation for re-planning, as well as re-deployment of applications. System adaptation by planners are therefore more suitable for significant changes in system workload, resources, objectives, and/or operating conditions that occur at lower frequency. Conversely, RACE employs control-theoretic adaptive resource management algorithms that performs system adaptation and have minimal or low overhead. System adaptation by RACE is thus more suitable for more frequent fluctuations in workload and resources, such as changes in application resource utilization and minor disturbances from external sources.

As a part of our ongoing work, we are empirically validating and evaluating the stated advantages of integrating online planning (SA-POP) and an adaptive resource management framework (RACE).

References

[1] S. Sharma and S. Curtis, "Magnetospheric Multiscale Mission," in *Nonequilibrium Phenomena in Plasmas*. Springer Verlag, 2005, pp. 179–195.

[2] J. W. S. Liu, *Real-time Systems*. New Jersey: Prentice Hall, 2000.

[3] J. S. Kinnebrew, A. Gupta, N. Shankaran, G. Biswas, and D. C. Schmidt, "Decision-Theoretic Planner with Dynamic Component Reconfiguration for Distributed Real-time Applications," in *The 8th International Symposium on Autonomous Decentralized Systems (ISADS 2007)*, Sedona, Arizona, Mar. 2007.

[4] N. Shankaran, D. C. Schmidt, Y. Chen, X. Koutsoukous, and C. Lu, "The Design and Performance of Configurable Component Middleware for End-to-End Adaptation of Distributed Real-time Embedded Systems," in *Proc. of the 10th IEEE International Symposium on Object/Component/Service-oriented Real-time Distributed Computing (ISORC 2007)*, Santorini Island, Greece, May 2007.

[5] D. Suri, A. Howell, N. Shankaran, J. Kinnebrew, W. Otte, D. C. Schmidt, and G. Biswas, "Onboard Processing using the Adaptive Network Architecture," in *Proceedings of the Sixth Annual NASA Earth Science Technology Conference*, College Park, MD, Jun. 2006.

[6] S. Bagchi, G. Biswas, and K. Kawamura, "Task Planning under Uncertainty using a Spreading Activation Network," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 30, no. 6, pp. 639–650, Nov. 2000.

[7] C. Lu, X. Wang, and X. Koutsoukos, "Feedback Utilization Control in Distributed Real-time Systems with End-to-End Tasks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 16, no. 6, pp. 550–561, 2005.