Available online at www.sciencedirect.com
ScienceDirect
The Journal of Systems and Software 80 (2007) 984–996
The Journal of Systems and Software
www.elsevier.com/locate/jss

A multi-layered resource management framework for dynamic resource management in enterprise DRE systems

Patrick Lardieri [a], Jaiganesh Balasubramanian [b], Douglas C. Schmidt [b,*], Gautam Thaker [a], Aniruddha Gokhale [b], Thomas Damiano [a]

[a] Advanced Technology Labs, Lockheed Martin Corporation, Cherry Hill, NJ 08002, United States
[b] Department of Electrical Engineering and Computer Science, Vanderbilt University, Nashville, TN 37235, United States

Available online 11 December 2006

Abstract

Enterprise distributed real-time and embedded (DRE) systems can benefit from dynamic management of computing and networking resources to optimize and reconfigure system resources at runtime in response to changing mission needs and/or other situations, such as failures or system overload. This paper provides two contributions to the study of dynamic resource management (DRM) for enterprise DRE systems. First, we describe a standards-based multi-layered resource management (ARMS MLRM) architecture that provides DRM capabilities to enterprise DRE systems. Second, we show the results of experiments evaluating our ARMS MLRM architecture in the context of a representative enterprise DRE system for shipboard computing.

Keywords: Dynamic resource management; Enterprise DRE systems; QoS-enabled component middleware

1. Introduction

Enterprise distributed real-time and embedded (DRE) systems, such as shipboard computing environments (Schmidt et al., 2001), airborne command and control systems (Loyall et al., 2001), and intelligence, surveillance and reconnaissance systems (Sharma et al., 2004), are growing in complexity and importance as more computing devices are networked together to help automate tasks previously done by human operators. These types of systems are characterized by stringent quality-of-service (QoS) requirements, such as low latency and jitter, expected in real-time and embedded systems, as well as high throughput, scalability, and reliability expected in enterprise distributed systems.

Enterprise DRE systems have a range of QoS requirements that may vary at runtime due to planned (Li and Nahrstedt, 1999) and unplanned (Abdelzaher et al., 2003) events. Examples of planned events include mission goal changes due to refined intelligence and planned task-completion exceeding mission parameters. Likewise, examples of unplanned events might include system runtime performance changes due to loss of resources, transient overload, and/or changes in algorithmic parameters (such as modifying an air threat tracking subsystem to have better coverage).

Dynamic resource management (DRM) (Welch et al., 1998, 2001) is a promising paradigm for supporting different types of applications running in enterprise DRE system environments—as well as to optimize and reconfigure the resources available in the system to meet the changing needs of applications at runtime. The primary goal of DRM is to ensure that enterprise DRE systems can adapt dependably in response to dynamically changing conditions (*e.g., evolving multi-mission priorities*) to ensure that computing and networking resources are best aligned to meet critical mission requirements. A key assumption in DRM technologies is that the levels of service in one dimension can be coordinated with and/or traded off against the levels of service in other dimensions to meet

* Corresponding author. Tel.: +1 615 2949573; fax: +1 615 3437440.
E-mail address: d.schmidt@vanderbilt.edu (D.C. Schmidt).

0164-1212/$ - see front matter
doi:10.1016/j.jss.2006.09.042

mission needs, *e.g.*, the security and dependability of message transmission may need to be traded off against latency and predictability.

This paper describes a *multi-layer resource management* (ARMS MLRM) architecture we developed to demonstrate DRM capabilities in a shipboard computing environment. This environment consists of a grid of computers that manage many aspects of a ship's power, navigation, command and control, and tactical operations (Schmidt et al., 2001) using standards-based DRM services that support multiple QoS requirements, such as survivability, predictability, security, and efficient resource utilization. Our ARMS MLRM was developed for the DARPA's *Adaptive and Reflective Middleware Systems* (ARMS) program (`dtsn.darpa.mil/ixodarpatech/ixo_FeatureDetail.asp?id = 6`), which is applying DRM technologies to coordinate a computing grid that manages and automates many aspects of shipboard computing. We describe and empirically evaluate how the ARMS MLRM manages computing resources dynamically and ensures proper execution of missions in response to mission mode changes and/or resource load changes and failures, as well as capability upgrades.

2. The ARMS multi-layered resource management middleware

This section describes the design and functionality of the component middleware used to implement the ARMS MLRM architecture.

2.1. ARMS MLRM design goals

Providing effective DRM capabilities for enterprise DRE systems depends on several factors that span the domain- and solution-space. For example, solutions to domain-specific issues, such as adapting to mission mode changes, capability upgrades or resource failures, are impacted by the choice of technologies and platforms used in the solution space. Addressing the complex problem of DRM as a single unit, however, can become intractable due to the *tangling of concerns* across the domain- and solution-space. Hence, there is a need to address DRM problems at different levels of abstraction, yet maintain a coordination between these levels. These design goals motivate the ARMS MLRM framework described in this section, which we applied to help resolve key DRM challenges in the ARMS program.

The goals of the ARMS MLRM design are to provide DRM solutions when missions change, resources fail or become available, failures occur due to damage, or new capabilities are added to the system. We addressed the following types of DRM problems to meet the needs of enterprise DRE systems:

- *Mission mode changes*, where the goal is to enable a much broader set of resource reallocations beyond mode changes and behaviors typically provided by domain applications. Meeting this goal requires the ARMS MLRM to determine *at runtime* which components should actually run in response to mission mode changes. Moreover, the ARMS MLRM must tune application performance parameters dynamically using increasingly finer-grained precision, as opposed to a coarse-grained, discrete set of configurations.
- *Load changes*, where the goal is to tolerate *out-of-spec* variations gracefully. The ARMS MLRM therefore ensures that available resources are allocated to the *currently most important* mission capabilities, allowing the system to scale gracefully to even *out-of-spec* loads.
- *Resource changes*, where the goal is to restore *all* mission capabilities, even those that were not designed to tolerate specific failures. Through the automatic recovery from such failures, the ARMS MLRM maximizes resource utilization and performance, while simultaneously maximizing the value of mission capabilities. In addition, we wanted to expand mission capabilities as new resources become available due to requirement or environment changes.
- *Capability upgrades*, where the goal is to allow capabilities to be dynamically introduced into the system that were not planned initially. We wanted to capture the general behavior of the newly introduced artifacts so the ARMS MLRM can determine the resource requirements of these artifacts dynamically and make appropriate allocations. In particular, the ARMS MLRM should be able to make dynamic resource allocation decisions for the newly introduced artifacts, even when such capabilities were not part of the original system.
- *Platform upgrades*, where the goal is to support the *heterogeneous* environment in which long-lived enterprise DRE systems operate. This environment involves diverse middleware, operating systems, CPUs, and networks.

The ARMS MLRM services are implemented as a set of common and domain-specific middleware services described in Section 2.2 that communicate using the standard middleware described in Section 2.3, which coordinates and encapsulates a wide range of operating systems, networks, programming languages, and hardware.

2.2. The component-based ARMS MLRM design

The ARMS MLRM architecture integrates resource management and control algorithms enabled by standard component middleware infrastructure described in Section 2.3 to achieve the enterprise DRE system challenges described in Section 1 and the ARMS MLRM design goals described in Section 2.1. These design goals, combined with the complexity of the enterprise DRE system domain, led us to model the DRM solution space as a layered architecture comprising components at different levels of abstraction. Fig. 1 depicts the layers in the ARMS MLRM,