Resolving Component Deployment & Configuration Challenges for Enterprise DRE Systems via Frameworks & Generative Techniques

Gan Deng
Department of EECS
2015 Terrace Place, Nashville, TN, 37203 USA
gan.deng@vanderbilt.edu

ABSTRACT

Component-based software engineering (CBSE) is increasingly being adopted for large-scale software systems, particularly for enterprise *distributed real-time and embedded* (DRE) systems. One of the most challenging—and often most neglected—problems in CBSE for enterprise DRE systems is the system (re)deployment and (re)configuration (D&C) process, where the increasing heterogeneity and versatility of application domains requires supports for an unprecedented level of configurability and adaptability. Existing D&C technologies suffer from two major problems: (1) insufficient module-level reusability and ability to evolve in the face of functionality evolution and diversification due to the interaction of too many orthogonal concerns imposed by a wide range of application requirements and (2) significant inherent and accidental complexities stemming from inadequate design tools. To address these problems, my research focuses on improving both computing performance and human productivity associated with the D&C of component-based enterprise DRE systems. To improve computing performance, my research has systematically identified bottlenecks with conventional D&C approaches and provides an aspect-oriented approach to decouple "extrinsic" orthogonal D&C concerns from "intrinsic" core D&C infrastructure, thereby enabling different crosscutting D&C concerns to be weaved independently to create a light-weight, highly optimized and extensible D&C infrastructure. To improve human performance, my research provides model-driven tools and analysis techniques to alleviate key inherent and accidental complexities in the D&C process.

Categories and Subject Descriptors

D.2.2 [Software Engineering]: Design Tools and Techniques: Modules and Interfaces; [Special-purpose and Application-based Systems]: Distributed Systems;

General Terms

Design, Performance

Keywords

Component-based Software Engineering, Component Middleware, Deployment and Configuration (D&C)

1. PROBLEM STATEMENT

Component based software engineering (CBSE) is a promising paradigm for achieving systematic reuse and composition of software artifacts [1]. Component middleware platforms, such as the CORBA Component Model (CCM), Enterprise Java Beans (EJB), and .NET, reify the CBSE paradigm and offer powerful abstractions, modularity, and expressivity for complex and heterogeneous computing environments. They also facilitate the development of high quality distributed applications with a shorter development cycle and reduced implementation efforts by separating component functional aspect from component systematic aspects, such as component lifecycle management. Component middleware systems are being adopted in a broad range of application domains, ranging from traditional enterprise platforms to mobile devices, embedded systems, real-time systems, and even safety-critical systems.

Although component middleware provides a higher layer of abstraction to build large-scale systems by composing reusable components, the system D&C process remains a challenging —*and often most neglected* — problem, particularly for enterprise *distributed real-time and embedded* (DRE) systems, such as total ship computing environments, real-time inventory tracking for warehouse management, and power grid control, where the right answer delivered too late becomes the wrong answer. Since DRE systems have stringent quality of service (QoS) requirements wrt latency, jitter, scalability, and resource usage, their D&C processes are typically complex and multi-faceted, e.g., their lifecycles involve system assembly specification, installation, QoS configuration, planning, QoS adaptation, and runtime redeployment and reconfiguration [7]. Unfortunately, existing D&C technologies suffer from the following problems:

1. Diversified, convoluted application D&C requirements result in heavy-weight, monolithic D&C infrastructure that adversely affects computing performance at DRE system deployment runtime. The design of reusable D&C frameworks for component middleware platforms is hard because their architectures should not be coupled with any particular application domain, operating environment, or hardware platforms. Hence, existing D&C implementations provide general-purpose solutions that deliver all the configurable modules to the system designers and deployers, thereby the logic of orthogonal modules are intermingled with each other and with that of the primary functionality. Moreover, since various D&C concerns tend to crosscut different layers of the software platforms, ranging from application components to component servers and containers to operating systems and hardware, it is impossible, by using traditional methods, to completely decouple them from one another based on particular system deployment and configuration scenario.

ICSE'06, May 20-28, 2006, Shanghai, China.
ACM 1-59593-085-X/06/0005.

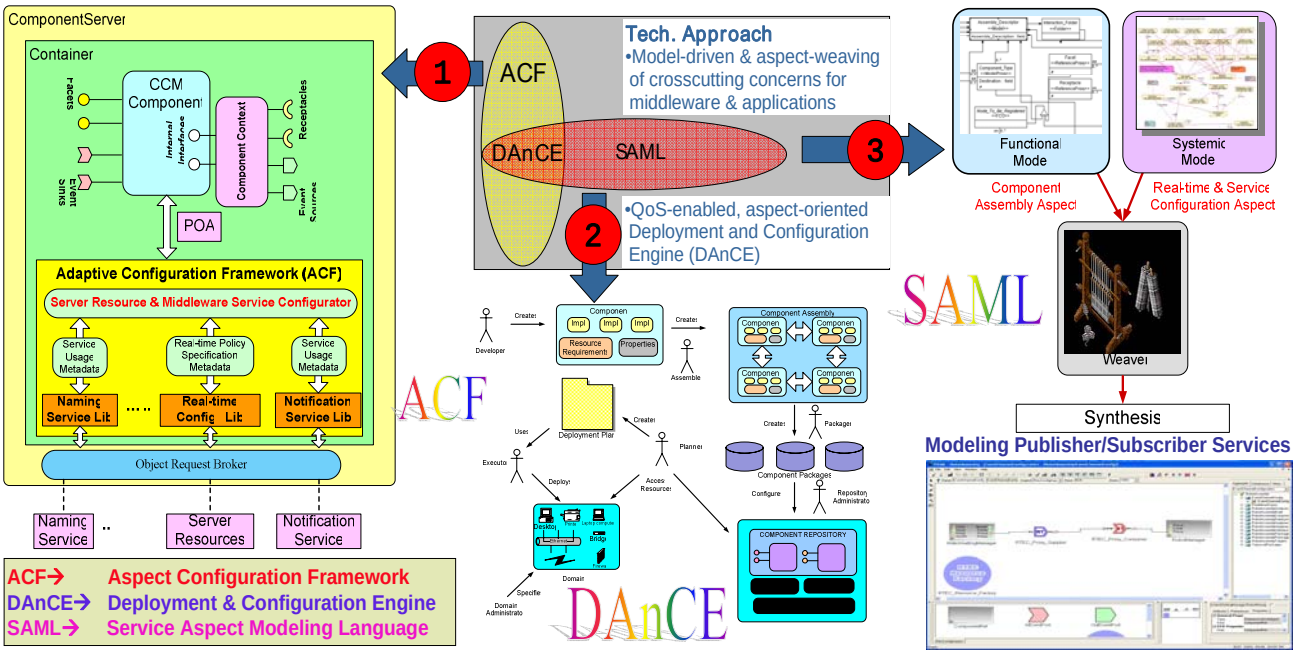

Figure 1: High-Level Structure of DynamicCIAO

2. Lack of QoS assurance and adaptability that adversely affects computing performance at DRE system runtime. Existing D&C frameworks that support (re)configuration and (re)deployment of component middleware platforms are not well-suited for enterprise DRE systems due to their inability to support stringent QoS requirements. For example, when component-based DRE systems are deployed into their target environment, conventional middleware D&C platforms do not ensure that QoS properties essential for DRE systems are met. Moreover, conventional D&C platforms do not maintain the stability and QoS of enterprise DRE systems during (re)deployment and (re)configuration of components in running DRE systems.

3. Manual, ad hoc techniques for deploying and configuring middleware QoS and middleware service aspects that adversely affect human productivity for developing component-based DRE systems. Conventional middleware D&C platforms are poorly suited for assembling DRE systems from pre-existing components that must meet stringent QoS and service configuration requirements. For example, these D&C platforms do not separate real-time QoS concerns (such as component server threading and priority models) and middleware service configuration concerns (such as component discovery and publish/subscribe integration) from application business logic and lifecycle management. It is therefore hard to reuse components for DRE systems, and the onus is on humans to manage these crosscutting concerns manually via ad hoc techniques, which impedes productivity and quality.

2. PROPOSED SOLUTIONS

The goal of my research is to address the R&D challenges described in Section 1 by developing and validating advanced middleware framework and generative software techniques that can significantly improve human productivity and computing performance associated with designing and executing the D&C aspects of component-based DRE systems. To achieve my goals I am systematically pinpointing the performance bottlenecks of existing D&C mechanisms for both initial system D&C and runtime system redeployment and reconfiguration, identifying the aspects that affect D&C performance, and then applying optimization techniques and generative techniques to alleviate these bottlenecks. In particular, through rigorous performance evaluation, I am exploring techniques to (1) identify and separate various orthogonal but crosscutting D&C concerns (e.g., real-time QoS concerns and middleware service configuration concerns) that adversely affect D&C performance from core deployment and configuration infrastructure, (2) allow these D&C concerns to be metaprogrammable and specified declaratively through higher-level metadata, (3) allocate the necessary execution environment resources as an integral part of the D&C process so the entire DRE system can meet end-to-end real-time QoS requirements, and (4) automatically (re)configure and manage the lifecycles of middleware services as an integral part of the D&C process. As a result, the primary functionality of the D&C is supported with a much simpler architecture and enhanced performance. Moreover, customization and configuration of the middleware for a wide-range of requirements becomes possible by weaving various concerns into the D&C infrastructure.

To help system deployers better deploy, configure and validate complex DRE systems, my research is developing an innovative methodology called *Aspect-oriented Model-based Configuration and Deployment process* (AMDaC), and build prototypes based on this methodology to evaluate and validate it empirically.

The high-level architecture of AMDaC is shown in Figure 1 and the dynamics and interactions among various modules within AMDaC are shown in Figure 2. As shown in Figure 1, the three key elements in the AMDaC methodology are:

- The ***Aspect Configuration Framework*** (ACF), which is a highly configurable and highly extensible middleware framework which uses aspect-oriented approach to weave specific D&C concerns.

- The ***Deployment And Configuration Engine*** (DAnCE), which is a distributed toolset that takes concrete D&C configuration description as input, and then drive the ACF to actually weave various D&C concerns into component middleware and component-based systems.
- The ***Service Aspect Modeling Language*** (SAML), which defines each D&C aspect as a separate view of the entire system, and use visual modeling language to model each concern, then the constraint checkers associated with SAML validate the models, and the code generators associated with SAML automatically synthesize the deployment and configuration descriptor files.

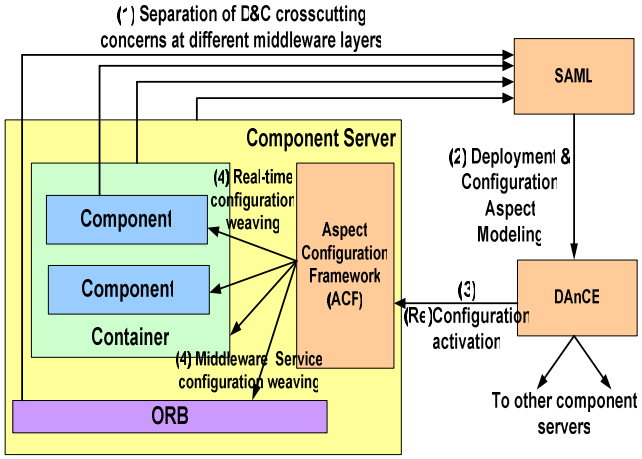

Figure 2: The Dynamics of the AMDaC Architecture

The AMDaC methodology addresses the R&D challenges described in Section 1 as follows:

1. Defining real-time QoS aspects and middleware service configuration aspects as first-class objects. In large-scale component-based DRE systems, two key D&C concerns involve (1) allocating system resource to components to ensure end-to-end system performance and (2) configuring and deploying components to use the underlying middleware services to achieve certain mission goals. To effectively enforce real-time QoS and middleware service configuration requirements in DRE systems to improve computing performance based on particular deployment and configuration scenario we treat these concerns as "extrinsic" concerns and decouple them from the "intrinsic" core D&C infrastructure. Additional design-time tools (such as SAML) and run-time tools (such as DAnCE) can then be used to simplify the configuration of such D&C concerns by addressing both inherent and accidental complexities associated in the D&C process. Since both real-time QoS aspects and middleware service configuration aspects are highly configurable, AMDaC raises the abstraction level of these aspects by defining them as first-class elements in the D&C infrastructure and allowing them to be specified declaratively via high-level configuration languages. We implement these capabilities in the *Aspect Configuration Framework* (ACF), which (1) provides an integrated set of fine-grained domain-specific structures and functionality based on patterns to configure real-time QoS and middleware services and (2) separates different D&C concerns of component-based DRE systems so each concern can be manipulated as independently as possible from other concerns. The novelty of ACF stems from the aspect-oriented techniques it uses to decouple orthogonal D&C concerns from core component middleware D&C infrastructure. These concerns can be woven into different layers of middleware and application components at the most appropriate time, i.e., during system deployment and at runtime, thereby resulting in a highly optimized configuration framework with much better QoS than conventional approaches.

2. Use generative techniques to weave crosscutting real-time QoS concerns and middleware service configuration concerns. Although ACF provides a set of highly configurable structures for configuring application and middleware components, these configuration options tend to crosscut multiple modularity boundaries – including different layers of middleware and multiple stages of the DRE system lifecycle, such as compilation, deployment, and run-time (re)configuration. AMDaC therefore applies generative aspect-oriented techniques to detangle various configuration concerns of application and middleware components from their functionality. To help DRE system developers deploy and configure both the component middleware and component-based applications, we developed the *Deployment And Configuration Engine* (DAnCE) [8], which is an aspect-oriented runtime framework that allows different D&C concerns, such as component server resource configuration, middleware service configuration, and component assembly reconfiguration, to be specified through higher-level metadata. DAnCE then guides the *ACF* to generate and weave these crosscutting concerns into different DRE system modules, e.g., application components, containers and component servers. The novelty of DAnCE stems from its generative ability to integrate different D&C modules dealing with different D&C concerns together, both during the initial system deployment and during the run-time redeployment and reconfiguration phase, thereby improving system QoS assurance and adaptability.

3. Simplify the D&C process and improve human productivity through generative model driven development (MDD) techniques. To simplify the development of component-based DRE systems, AMDaC provides a generative model-driven development (MDD) tool called the *Service Aspect Modeling Language* (SAML), which is a *domain-specific modeling language* (DSML) built with a metamodeling toolkit called the *Generic Modeling Environment* (GME) [3]. SAML supports the configuration, deployment, and validation of component middleware and applications, and it exploits the metamodeling approach to provide a multi-aspect, multi-view modeling environment to allow system developers to (1) model DRE system resource usages and real-time QoS policies and (2) specify middleware service behaviors and configuration requirements. SAML enables visual manipulation of modeling elements and performs various generative actions, such as synthesizing (1) XML-based deployment and configuration descriptors and (2) middleware service-specific configuration files. The novelty of SAML is its use of generative MDD techniques to shield DRE system developers from writing tedious and error-prone XML-based deployment and configuration descriptors manually to improve human productivity.

3. EVALUATION

The contributions of my research include: (1) developing and validating a new design methodology called AMDaC to address key D&C concerns to improve computer performance and human productivity for component-based DRE systems, (2) building frameworks and tools based on my methodology to empirically evaluate the pros and cons of this methodology compared with

conventional component-based system D&C approaches, and (3) capturing key design patterns associated in building a deployment and configuration infrastructure in the form of reusable frameworks and generative tools.

I plan to validate my methodology, techniques and tools in the context of the OMG defined D&C specification [7] and based on *Component Integrated ACE ORB* (CIAO) [2] Lightweight CCM [9] component middleware implementation since it is a representative example of QoS component middleware for DRE systems. The SAML modeling tool is built with GME metamodeling toolkit [3]. The evaluation criteria will be based on the empirical performance enhancement measurement and productivity enhancement measurement compared with existing D&C frameworks. The frameworks and tools developed in my research will be used a vehicle to validate my techniques, however, my research is applicable to general component D&C processes and can be applied to other component models, as well, such as the EJB component model and Prism [11] component model.

4. RELATED WORK

Recently, to simplify the deployment and configuration of component-based systems, OMG has standardized this process and announced the OMG Deployment & Configuration (D&C) specification [12]. The goal of this specification is to promote component reuse and allow complex application to be built by assembling existing components. However, to provision dynamic adaptation capabilities to meet desired QoS to accommodate changing environment remains a challenging problem, particularly for enterprise DRE systems, since these systems have stringent quality of service (QoS) requirements, and my research enhances the OMG D&C Specification to address this problem.

Researchers have applied generative software development approach to middleware architectures for different purposes and in different ways. The QuO project at BBN Technologies (quo.bbn.com/) constitutes a framework supporting the development of distributed applications with QoS requirements. QuO uses quality description languages (QDL) to specify client-side QoS needs, regions of possible level of QoS, system conditions that need to be monitored, and QoS conditions. Loyall et al. [6] interpret these different description languages as aspect languages that are processed by a code generator to build a runtime environment to support the desired QoS by client and server in a distributed application. AMDaC differs from QuO by focusing on complex, multi-layered architectures that uniquely address key challenges associated in the component-base software engineering, whereas QuO focuses on the QoS issues of traditional client/server models.

Zhang et al. [5] use aspect-oriented techniques to improve the customizability of the middleware core infrastructure at the Object Request Broker (ORB) level. The crosscutting problems they address are generally applied to orthogonal functionality of object-oriented middleware. In contrast, AMDaC addresses key challenges for component-based software systems, particularly for large-scale DRE system.

5. CURRENT STATUS

The DAnCE framework, which is one significant part of my thesis, has been designed and implemented. A paper describing DAnCE [8] was presented at the Component Development 2005 conference. Another key part of my research is the *Event QoS Aspect Language* (EQAL) (a prototype of SAML), which has been designed and implemented using generative model-driven techniques to automate many service configuration and deployment tasks associated with integrating publish/subscribe services into QoS-enabled component-based systems. Two papers describing EQAL [4, 10] were presented at the ACM Generative Programming and Component Engineering (GPCE) 2004 conference and ACM/IEE MoDELS 2005. The ACF framework is in its initial design and implementation phase. Other parts of my thesis are under various stages of maturity. For details about my publication record and research background, please see www.dre.vanderbilt.edu/~dengg.

6. REFERENCES

[1] G.T. Heineman, B.T. Councill, "Component-Based Sofware Engineering: Putting the Pieces Together", *Addison-Wesley*, Reading, Massachusetts (2001).

[2] N. Wang, D.C. Schmidt, A. Gokhale, C. Rodrigues, B. Natarajan, J.P. Loyall, R.E. Schantz, and C.D. Gill, "QoS-enabled Middleware", *Middleware for Communications*, Wiley, NY, 2003.

[3] G. Karsai, J. Sztipanovits, and A. Ledeczi,"Model-Integrated Development of Embedded Software", *Proceedings of the IEEE*, volume 91, number 1, pages 145-164, Jan. 2003

[4] G. Edwards, G. Deng, D. Schmidt, A. Gokhale, and B. Natarajan, "Model-driven Configuration and Deployment of Component Middleware Publish/Subscribe Services", *Proceedings of the 3rd ACM International Conference on GPCE*, Vancouver, CA, October 2004.

[5] C. Zhang, H-A. Jacobsen: "Resolving Feature Convolution in Middleware Systems" *Proceedings of the 19th Annual ACM SIGPLAN Conference on OOPSLA*, October 24-28, 2004, Vancouver, BC, Canada.

[6] J.P. Loyall, D. Bakken, R.E. Schantz, "QoS Aspect Languages and Their Runtime Integration", *Lecture Notes in Computer Science, vol. 1511*, Springer, 1998.

[7] Object Management Group, "Deployment & Configuration Specification", July, 2003, ptc/2003-07-08.

[8] G. Deng, J. Balasubramanian, W. Otte, D. Schmidt, and A. Gokhale, "DAnCE: A QoS-enabled Component Deployment and Configuration Engine", *Proceedings of the 3rd Working Conference on Component Deployment*, Grenoble, France, November 28-29, 2005.

[9] Object Management Group: "Lightweight CORBA Component Model Revised Submission", *Object Management Group, Inc.* May 2003, realtime/03-05-05.

[10] G. Deng, G. Lenz, D.C. Schmidt, "*Addressing Domain Evolution Challenges in Model-Driven Software Product-line Architectures Addressing Domain Evolution Challenges for Model-driven Software Product-line Architectures (PLAs)*", Proceedings of the ACE/IEEE MODELS 2005 Workshop, October 2, 2005, Jamaica.

[11] D.C. Sharp, "Reducing Avionics Software Cost Through Component Based Product Line Development", *Proceedings of the 10th Annual Software Technology Conference*, Salt Lake City, Utah, April 1998.

[12] Object Management Group, "Deployment & Configuration Specification", July, 2003, ptc/2003-07-08.