Experiences with Middleware for a Networked Embedded Software Technology Open Experimental Platform

Venkita Subramonian and Christopher Gill
Department of Computer Science, Washington University, St. Louis, MO
{venkita, cdgill}@cs.wustl.edu

Abstract[1]

Networked Embedded Software Technology (NEST) [1] defines a class of Distributed Real-time Embedded (DRE) systems in which the number of computing nodes is large, typically between 10^2 and 10^5 nodes, and resources per node are highly constrained. With the advent of small, robust, and inexpensive Micro-Electro-Mechanical Sensors (MEMS) and actuators, computing nodes of this scale are being used increasingly in fields like advanced avionics and space systems. This has prompted the need for collocation and embedding of processing and control software within or in close proximity to MEMS components with strict memory and processing constraints

NEST systems involve design forces from a number of key areas which must be resolved at once – *e.g.*, distribution of control, heterogeneous processing, time synchronization, resource allocation, group membership and voting, fault tolerance, and system level configuration management. Middleware approaches, such as Common Object Request Broker (ORB) Architecture (CORBA) [2] ORBs can serve to encapsulate the inherent complexity of these design forces. However, middleware solutions for NEST appear highly dependent on the particular application. To address the specific design forces for each application, we therefore take a two-fold approach: we will (1) identify and apply patterns and pattern languages for NEST middleware, and (2) use advanced generative [3] programming techniques to re-factor the middleware frameworks that reify these patterns, for maximum flexibility.

Patterns we have identified so far as candidates for NEST middleware [4] include *Lookup* [5], *Leasing* [6], *Lazy Acquisition* [7], *Blackboard* [8], *Broker* [8], *Distributed Observer* [9], *Distributed Scheduler* [9], *Replication Manager* [10], *Fault Detector* [10], and *Fault Notifier* [10]. We will examine these patterns for applicability in the NEST environment, and generate both a pattern catalog and generative pattern languages for specific NEST application sub-domains. We also plan to mine patterns in other related technology areas, *e.g.,* Jini [11], JavaSpaces [12], the Real-Time Specification for Java [13], and the OMG Smart Transducers RFP [14] and response [15].

Our presentation will provide a snapshot of the design patterns and pattern languages that have been applied or that we plan to apply to a middleware framework we are developing for the Boeing NEST Open Experimental Platform (OEP) [4], and their implications for NEST systems. Furthermore, we will discuss progress to date in applying advanced programming techniques to re-factor and subset features of The ACE ORB (TAO) [16] to reduce its memory footprint within the design constraints of the Boeing NEST OEP.

[1] This work is funded under subcontract to Boeing by DARPA contract F33615-01-C-1898

References

[1] DARPA BAA #01-06, *Networked Embedded Software Technology (NEST)*, www.darpa.mil/ito/Solicitations/CBD_01-06.html

[2] Object Management Group, *The Common Object Request Broker: Architecture and Specification*, OMG document formal/2001-02-33

[3] Czarnecki and Eisenecker, *Generative Programming*, Addison-Wesley, 2000

[4] Subramonian, Gill, and Sharp, *Towards a Pattern Language for Networked Embedded Software Technology Middleware*, OOPSLA 2001 Workshop 11, “Towards Patterns and Pattern Languages for OO Distributed Real-Time and Embedded Systems”.

[5] M. Kircher, and P. Jain, *Lookup*, European Pattern Languages of Programs conference, Kloster Irsee, Germany, July 5-9, 2000.

[6] P. Jain and M. Kircher, *Leasing*, Pattern Languages of Programs conference, Allerton Park, IL, Aug. 2000

[7] M. Kircher, *Lazy Acquisition*, European Pattern Languages of Programs conference, Kloster Irsee, Germany, July 4-8, 2001.

[8] F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad and M. Stal, *Pattern-Oriented Software Architecture—A System of Patterns*, Wiley, 1996.

[9] N. Islam and M. Devarakonda, *An Essential Design Pattern for Fault-Tolerant Distributed State Sharing*; Communications of the ACM 39(10), 1996, pp. 65 – 74

[10] Object Management Group, *Fault Tolerant CORBA Specification*, OMG Document orbos/99-12-08.

[11] Sun Microsystems, *Jini Network Technology*, www.sun.com/jini/

[12] Sun Microsystems, Inc., *JavaSpaces Technology* java.sun.com/products/javaspaces/index.html

[13] Java Community Process, JSR 1, *Real-Time Specification for Java*

[14] Object Management Group, *Smart Transducers RFP*, orbos/00-12-13, Dec 15, 2000

[15] Object Management Group, *Smart Transducers Interface*, orbos/2001-06-03, Jun 18, 2001

[16] *The ACE ORB (TAO)*, www.cs.wustl.edu/~schmidt/TAO.html

Filename: NESTOMG_Jan_02.doc
Directory: E:\MyDocuments\School
Template: C:\Documents and Settings\venkita\Application Data\Microsoft\Templates\Normal.dot
Title: How NEST can be considered as a special kind of DRE system
Subject:
Author: venkita
Keywords:
Comments:
Creation Date: 8/29/2001 5:53 AM
Change Number: 5
Last Saved On: 8/29/2001 6:06 AM
Last Saved By: Chris Gill
Total Editing Time: 5 Minutes
Last Printed On: 5/1/2002 2:36 PM
As of Last Complete Printing
Number of Pages: 2
Number of Words: 727 (approx.)
Number of Characters: 4,149 (approx.)