Application of Middleware and Agent Technologies to a Representative Sensor Network

John S. Kinnebrew*
john.s.kinnebrew@vanderbilt.edu

William R. Otte*, Douglas C. Schmidt*,
Gautam Biswas*, and Dipa Suri**

* Vanderbilt University
EECS Department & ISIS
Nashville, TN 37203

** Lockheed-Martin Space Systems
Advanced Technology Center
Palo Alto, CA. 94304

Supported by NASA ESTO (AIST-ROSES program)
Sensor Webs

- Embedded systems
  - Soft/hard real-time
  - QoS requirements
  - Limited computational resources
- Power management
- Distributed resources
- Intermittent communication
  - Temporary/ permanent loss of access to data
- Changing network topology
- Top-down and bottom-up forces affect utility of tasks/configurations
  - User requests provide goals for data collection and analysis (top-down)
  - Local conditions determine appropriate tasks to achieve goals (bottom-up)
Multi-agent Architecture for Coordinated Responsive Observations (MACRO)

- **Mission level**
  - Mission agent controls a sensor net
  - User agents provide interface for applications and scientists
  - Brokers mediate contract net negotiations

- **Resource level**
  - Exec agent in control of local resource group
  - Other agents as necessary
  - Component middleware infrastructure
SEAMONSTER Objectives

SouthEast Alaska MOnitoring Network for Science Technology Education and Research

- Scientifically Motivated
  - Technology Development funded by NASA ESTO (AIST)
- Path for Technology Infusion
  - Scientific Collaborations
- Testbed Sensor Web
  - Technology Collaborations
CORBA Component Model (CCM) – Overview

- Components encapsulate application “business” logic
- Components interact via ports
  - Provided interfaces, e.g., facets
  - Required connection points, e.g., receptacles
  - Event sinks & sources
- Attributes
- Containers provide execution environment for components with common operating requirements
- Components/containers can also
  - Communicate via a middleware bus &
  - Reuse common middleware services
**Deployment Infrastructure Overview**

- **Repository Manager**
  - Database of components that are available for deployment ("staging area")

- **Target Manager**
  - Retrieval of target data (i.e., available nodes & resources)

- **Execution Manager**
  - Execution of an application according to a "Deployment Plan"

- **Domain Application Manager**
  - Responsible for deploying an application at the domain level

- **Domain Application**
  - Represents a "global" application that was deployed across nodes

- **Node Application Manager**
  - Responsible for deploying a locality constrained application onto a node

- **Node Application**
  - Represents a portion of an application that’s executing within a single node

---

www.cs.wustl.edu/~schmidt/PDF/DanCE.pdf
System Constituents

- GME
- PICML
- CoSMIC
- CORBA Services
- TAO
- ACE
- ReDAC
- RACE
- DaNCE
- CIAO
- Application Deployment and Configuration
- MACRO Architecture
- Agents
- Contract Net
- SA-POP
- Model Driven Software Development
- Dynamic Resource Management
- Smart sensing, autonomy, adaptability
- Interoperability

Model Driven Software Development
Dynamic Resource Management
System Construction

- **ACE OS Hardware**, Level 1: portability for hardware, operating system
- **TAO Middleware**, Level 2 based on CORBA 2 specification: distributed objects, interoperability for distributed heterogeneity
- **CIAO Middleware**, Level 3: based on CORBA 3 specification; distributed components, system configuration, & deployment

**CORBA Services**
- **Naming Service**: Agent locator (white pages)
- **Event Service**: Agent Communication (publish/subscribe paradigm)

**SA-POP Planner** – Agent reasoning; used as a service

**MACRO Agents**
- Dynamic resource management & (re)configuration

**RACE**
- CORBA Services

**ReDaC**
- Dynamic resource management & (re)configuration

**CIAO**
- Middleware Level 3; based on CORBA 3 specification; distributed components, system configuration, & deployment

**TAO**
- Middleware, Level 2 based on CORBA 2 specification; distributed objects, interoperability for distributed heterogeneity

**ACE**
- Middleware, Level 1; portability for hardware, operating system

**OS**

**Hardware**
Integrated System

MACRO Agents
SA-POP  RACE  CORBA Services
ReDAC

MACRO Agents
SA-POP  RACE  CORBA Services
ReDAC

MACRO Agents
SA-POP  RACE  CORBA Services
ReDAC

CIAO (and DaNCE)

TAO

ACE
Linux

ACE
Windows

ACE
VxWorks
MACRO Testbed Hardware (1/2)

- Closely emulate SEAMONSTER environment
  - 2 Vexcel Microservers
  - 3 SLUGS w/ WET54G Wireless/Ethernet bridges
  - 10 Motiev tMote Sky

- Vexcel (Microsoft) Microservers
  - Low-power ARM Single Board Computers
  - Power Conditioning Subsystem
  - COTS Wi-Fi/Ethernet bridge
  - WiFi Signal Amplifier
  - GPS
  - Solar charging regulator
  - Weather/Cold/Bear-proof case
MACRO Testbed Hardware (2/2)

- **SLUGS**
  - Re-purposed Linksys NSLU2 Network-Attached-Storage
  - Low-cost ARM Single Board Computers
  - Communicate using WET54G Wi-Fi/Ethernet bridges

- **Moteiv tMote Sky**
  - Low-power field sensors
    - Temperature
    - Humidity
    - Light
  - 2.4 Ghz 802.15.4
  - USB connector for base station or external sensor
Testbed Topology w/ Physical Distribution

- Provides best fidelity to actual SEAMONSTER environment
- Most difficult in terms of connectivity
- Consists of at least three physical locations
  - Microserver with “weather station” & tMote network
  - Microserver which collates data from several SLUGS
    - Two of three SLUGS in different locations (with attached tMote networks)
Avoids potential firewall problems with physically separated layouts

Single tMote network due to ZigBee ad-hoc network protocol

μServer not directly connected to Server has a WAP (running in WDS mode)

Second ‘hop’ through second μServer may present communication challenges for deployment & configuration
Middleware Integration Challenges (1/3)

- **Context**: Adapting to changing network topology
  - Sensor networks are often deployed in remote/inaccessible locations
  - Limited resources and/or damage may induce temporary loss of communication with nodes

- **Problem**: Failed links or nodes cause temporary or permanent loss of access to data stored on effected nodes

- **Solution Approach**
  - Introduce asynchronous publish/subscribe ports into agent components deployed onto nodes
  - Agents publish noteworthy data to these ports, and log data received
  - Data peers managed by deployment infrastructure
Middleware Integration Challenges (2/3)

- **Context** –
  - Sensor nodes may be interested in large numbers of observable phenomena
  - Type, duration, and frequency of observation may change over time

- **Problem** – Limited resources (processor, bandwidth, storage) requires prioritization of observable phenomena

- **Solution Approach** –
  - Nodes contain components implementing agents capable of intelligent, autonomous planning
  - Agents may influence deployed applications through re-deployment interfaces and CCM component homes
Middleware Integration Challenges (3/3)

- **Context** –
  - Sensor nodes often have limited power, changing weather conditions may impede ability to re-charge batteries
  - Nodes may need to periodically power down to conserve battery life

- **Problem** –
  - Sleep/wake cycles causes the infrastructure and applications to lose state
  - Deployment infrastructure must preserve state to correctly re-deploy application
  - Application state must be preserved

- **Solution Approach** –
  - Describe all deployments as locality-constrained
  - Maintain entire deployment tool chains on each node
  - Periodically instruct agents to save state using CCM-defined `ccm_store` and `ccm_load` operations
Future Integration Challenges

- Resource Constraints
  - Sensor nodes have limited processing and memory
  - Relatively large footprint of CCM limits number of components
deployed to a single node
- Infrastructure Fault Tolerance
  - Uncertain and harsh nature of many sensor web environments
    presents substantial challenge to deployment infrastructure
  - Current solution unnecessarily coarse-grained and resource heavy
- Communication in Sparse Wireless Networks
  - Point-to-point communication is an implicit requirement of
    CORBA/CCM
  - Challenge currently avoided using infrastructure-based wireless
    networks
  - CORBA Wireless Access and Terminal Mobility specification may
    provide better solution
Questions and Discussion
Deployment Infrastructure Overview (2/2)

For each Node in the Deployment Plan:

1. Uses plan. Execute it in the target environment.
2. Plans deployment of application based on resource data from resourceDataProvider. Resolve the package using searchPath. Produce a compatible deployment plan.
3. Install and configure packages in repository.
4. Planner
   - +searchPath
   - +resourceDataProvider
5. DeploymentPlan
   - +searchPath
   - +uses
6. RepositoryManager
   - <<manage packages>>
   - +searchPath
   - +findPackage
   - +uses
7. Executor
   - +preparePlan
8. TargetManager
   - +commitResources()
   - +releaseResources()
   - +resourceDataProvider
9. DeploymentPlan
   - <<creates>>
10. NodeApplicationManager
    - +spawns
11. DomainApplicationManager
    - +preparePlan
12. NodeManager
    - +spawns
13. NodeApplication
    - 1..*
    - +spawns
14. DomainApplication
    - 1..*
    - +preparePlan
15. RepositoryAdministrator
    - <<manage packages>>
16. RepositoryManager
    - 1
    - +searchPath
    - +findPackage
    - +uses
17. Planner
    - <<creates>>
18. ExecutionManager
    - 1
    - +preparePlan
19. Executing Executor
20. Infrastructure (Services)
SA-POP & RACE in MACRO

- **SA-POP**
  - Dynamic planning and scheduling under uncertainty
  - Replanning/rescheduling
  - Domain knowledge captured in TaskNetwork and TaskMap

- **RACE**
  - Dynamic resource allocation
  - Control algorithms for maintaining required QoS
  - Pluggable allocation and control algorithms