

# Intelligent Coordinating Entities Based Control Software Design<sup>\*</sup>

S. Ramaswamy

Software Automation and Intelligence Laboratory

Department of Computer Science

Tennessee Technological University

Cookeville TN 38505

Phone: 931-372-3448

Email: [srini@acm.org](mailto:srini@acm.org) / [srini@ieee.org](mailto:srini@ieee.org)

**Abstract:** Software systems designed to be distributed, interactive and intelligent (in a domain-specific sense), with ubiquitous human interfaces and the ability to exhibit intelligent cooperative behaviors will pave the way for achieving realistic flexible automation systems. In specific, the research conducted through the VERAM (Virtual Environments for Robotics, Automation and Manufacturing) at the Software Automation and Intelligence Laboratory (SAIL) at Tennessee Technological University, investigates the applicability of agent-based design techniques in the development of interactive software components designed as ICE (Intelligent Coordinating Entities) elements.

## 1. Introduction

Most modern day, industrial-strength systems are complex and operate in a dynamically changing environment. Building software systems that are easier to use in such applications implies addressing the issues of increasing complexities in *managing processes and communication between processes*. Current system simulation and prototyping techniques, based on highly simplistic models, involve a costly, labor-intensive modeling and development process. In order to manage complexity, it is very important to have tools, notations and methodologies that support the designer's work during design resolution. Moreover, advances in virtual reality (VR) have made it feasible to directly utilize VR for the modeling and simulation in virtual environments. The use of virtual reality in simulating manufacturing control software environments give designers the opportunity to play a pro-active role in

identifying flaws and optimizing the design. Current techniques do not allow the designer to be progressively involved in the design process. Often, the designer is limited to the initial specification stages and her/his experience is not exploited throughout the design process.

Software entities, called Intelligent Coordinating Entities - or ICE, as proposed by this research, go a step further than intelligent agents. ICE elements extend the agent-oriented design approach to implicitly include a coordination framework implemented through different enabling technologies (such as DCOM/CORBA/JINI), which are necessary to bring such intelligent coordinated behaviors to fruition. In a system, an ICE element (ICICLE), therefore, not only consists of a representation, decision and reaction structure, but also includes the mechanisms to orchestrate such reactions in a coordinated fashion using a well-defined communication structure as the defining baseline for all ICICLES. Thus, a network-centric communication structure forms the core of every ICE element (similar to the hexagonal structure in the snow crystal analogy). Depending on the enabling technology to be used, it may or may not be necessary to include the code for the communication structure within each ICICLE. However, depending upon the other players in the coordination, an ICICLE can selectively choose the appropriate information it is willing to share with the group by means of a dynamic model of its currently available services.

## 2. ICE Element Design

A flexible communication structure forms the basis of any ICE element. A software system designed with

---

\* Several students in SAIL have contributed to, and benefited from, research activities at the SAIL laboratory at Tennessee Technological University. These include: (i) Thad Scalf, Joe Cherry and Karthigan Srinivasan: Naval Radar Simulation Project. (ii) Andrew Trent and William Holcomb: Self Organizing Software Entities Project. (iii) Matt Irwin, Noah Rosser, Derrick Muncy and : Home Automation Project. (iv) VijayAnand: Power Plant Distribution Visualization Project.

ICE elements, or ICICLES, assumes a networked structure with dynamically interacting ICECLES. The interaction capability of an ICECLE is time-variant and hence may be different at different time instances depending upon the composition of the system and the other elements involved in the interaction. Each ICECLE may or may not have a corresponding hardware resource associated with it - the assumption being that all hardware devices come with an inherent ability to communicate with other existing group elements. This communication capability is platform and language independent and hence can be brought forth by using any standard specifications such as CORBA [1], JINI[2], etc.

On top of the above communication framework, coordination capability is built by maintaining a dynamic "information-sharing" model within each element. This model is built / updated and used to verify the availability of a service / information by the requested ICICLE (server), when the service is requested by another ICICLE (client). This model is dynamically generated / updated at the server by exploding / imploding details on specific services depending on the interaction client. With the server providing such a service, the client will be able to assimilate information about the server and hence act either pro-actively or reactively within the software system.

For example, assume that this technique has been used to build a automated factory floor. For simplicity, assume that each factory floor resource, would therefore, in essence be an ICICLE. Assume that a resource, say R1, at the beginning of a job flow is stuck with a difficult job. With its communication structure, it is able to communicate this information to other members of the group if it chooses to do so. Correspondingly, these other members can reevaluate their goals / strategies by spending their resources more productively by reacting intelligently to this information. However, in designing such a system, the first order of priority is to devise a mechanism to discern the information to be shared! To accomplish this we will use a Petri-net based technique to extract all the decision-points in the model. The hierarchical Petri-net model with the capability for selective expansion / contraction of subnets allow for selective information sharing.

ICICLES are based on a flexible communication structure with incrementally built flexible coordination, reasoning and other application-specific modules built on domain-specific needs. Thus,

software systems can be built as a combination of components that vary in their capabilities ranging from ones that are highly intelligent to those that are dumb. Moreover, the dynamic cooperation model allows for components that can have a time-dependent interaction behavior. In a system with multitude of ICICLES, the presence or absence of a particular component would thus be of little significance as long as the system is build with redundant structures- if a component is present and is willing to cooperate it will be utilized. This can allow for the flexibility to leverage industry and vendor partnerships.

### 3. Current Projects at SAIL

#### 3.1. Self Organizing Software Entities

This project attempts to design simple software structures in JAVA using agent-based design concepts and KQML messages to build systems with interacting and coordinating structures. The project is focused on developing a prototype system consisting of different-sized boxes that cooperate to execute a command (or goal) - i.e. organizing into a given pattern such as a line, circle, etc. The experience gained from implementing this prototype is to be used to implement a flexible system consisting of group of machines with differing capabilities that can coordinate / cooperate among themselves to design and execute dynamic process schedules for a collection of jobs (or goals). A screen shot of the above project is shown in Figure 1.

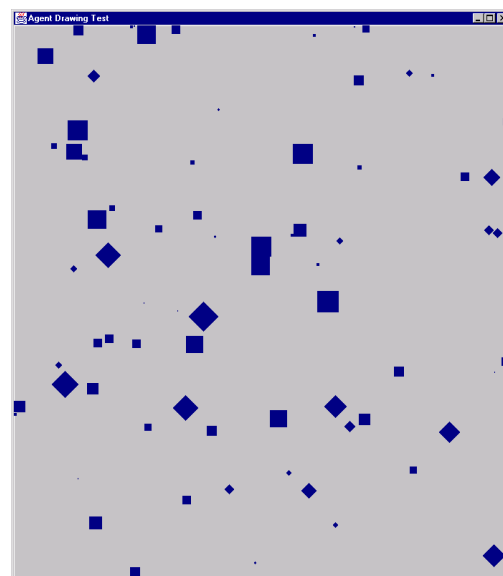


Figure 1. Self Organizing Structures

### 3.2. Home Automation

A Simple home automation scheduler has also been developed. It involves the virtual simulation of two simple home appliances - a coffeepot and a light that are programmed through a graphical interface. These appliances can be set to go off or on either at random or scheduled (either periodically or as a one-time operation at some specified time). A snapshot of this application is shown in Figure 2 - note the subtle shade difference in Figure 2b due to the light ON state. The application has integrated sound capabilities wherein the sound of the coffeepot in

operation diminishes as we move away from the appliance within the virtual world.

### 3.3. Naval Radar Simulation

A two-level distributed simulation environment for interference detection and frequency assignment in naval radar units has been developed. The Java-based simulation is designed as a two level architecture incorporating *i*) multiple radars on each ship controlled through a lower-level intra-ship interference control module and, *ii*) multiple ships in a group coordinated through a higher-level



a. OFF State



b. ON State (auto-timer driven)

Figure 2. A Simple Home Automation Prototype

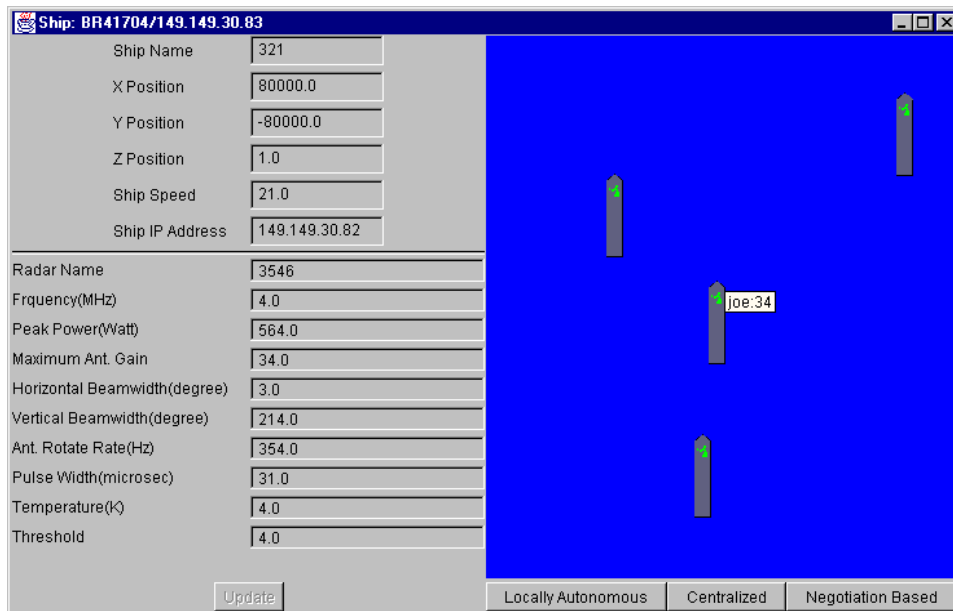


Figure 3. Lower-Level Control Agent Interface

interference control module. Each ship's interference detection and control mechanism is composed of these two separate levels, called a control agent, dynamically coordinating with one another in eliminating interference problems. In addition, the control agent is a multi-threaded architecture that incorporates the maintenance of a distributed data base system that contains periodically updated radar information. Current implementation involves centralized, locally autonomous and negotiation-based interference resolution strategies. A screen shot of the interference resolution process is shown in Figure 3.

### 3.4. Power Plant Distribution Visualization

Operator-visualization of observable data in a power system control center will enable efficient and reliable decision-making, especially under stressful conditions. Open access environments and the ever-increasing crunch on generation capacity augmentation are creating unprecedented demand for the availability of more "processed" information to help in the selection of safe and economically acceptable options. Recent advances in computing technology have forced data presentation to a power system control center operator, to evolve from simple one-line diagrams to animations and 3-D environments. This recently begun research project will strive to develop a visualization package for a power system control center. The prototype will focus on functions such as: available capacities of generation and transmission, voltage levels, and some financial cost information. Data for this prototype can be gathered from individual databases or from analysis programs making on-line calculations. The underlying philosophy will be to minimize the use of costly and generalized software packages, and to incorporate ergonomic and human-centric design principles. The project will exploit advances in virtual reality and multimedia techniques for interactive design, analysis and simulation. Such virtual environments offer the ability to evaluate current capabilities, future plans and schedules, without the investment of vast technical and financial resources. As an experimentation tool, the interactive virtual environment will allow power companies to conduct a rapid and thorough assessment of risks, affordability, ease of production, and other impacts on power generation capabilities. Moreover, simulation in the virtual environment will provide a

foundation for continued experimentation and process development, thereby aiding market advancement. Finally, simulated training in such virtual environments promises to be a useful approach to provide operators the necessary competitive advantage. The prototype software will focus on the human-centric design principles to aid the operator in navigating through complex virtual environments. Specifically the capabilities will include:

1. **Rapid Navigation Capability to Reduce Navigational "Stress" on the Operator:** The package will provide the capability for rapid navigation through three-or-four menu or button-controlled customized views (top, bottom, side-view, etc.) of the control center data that would ease the navigational "stress" on the operator. In each such view, the operator will have the capability to select grid lines both individually, or as a group and the ability to zoom to specific locations. In such a virtual environment with rapid navigational facilities, the operator has the obvious advantage of having *greater time to concentrate on issues that boost productivity and not spend their time on issues that induce stress and fatigue.*
2. **Exploit Language-Specific Performance Capabilities:** The software will be multi-language based - it will derive from the capabilities of the Virtual Reality Modeling Language (or VRML) to create and visualize a virtual environment, while it will use the logic and processing capabilities of Java to interact with the VRML created virtual environment. The advantage to this approach is that both VRML and Java are "good" at what they do and do it well with respect to performance considerations.
3. **Seamless Environment to Reduce Hardware Capability Restrictions:** Both VRML and Java are *neither platform-specific nor location-specific* languages, and hence can provide a seamless virtual environment without specific hardware/software demands. The environment can be implemented on a multitude of machines and execute satisfactorily on quite a few widely used operating systems. It can be built with the ability to read VRML data, thereby **allowing the**

**sharing of virtual worlds** created by other software packages.

4. **Licensing and Other Indirect Costs:** Finally, such a virtual environment will be very cost-effective in terms of hardware and software acquisition, as well as their maintenance. A network of desktop machines with mostly public domain software can host a huge database of generation and distribution data. Also, the machines can be serviced much more cheaply through local information technology departments. Moreover, Java will provide the necessary interoperability through Java Database Connectivity (JDBC) calls to any database that holds generation and distribution data. Due to its network-centric development, it provides better features for local area network operations and database connectivity.

#### 4. **Expected Research Impact**

The ICE based approach will lead to the development of a comprehensive design framework for developing flexible, interactive, distributed software systems, with domain-specific intelligence. The research will provide a new thrust and direction for the development of control software systems. A potential payoff will be the efficient and flexible control strategies that may be easily derived from the design efforts. Although the advent of software based control has greatly improved our manufacturing sectors, much of our current day woes can also be easily attributed to this inherent inflexibility provided by software based control. However, this problem is not solely restricted to this domain. Often, the problem is because the software is not effectively designed and tested. A simple by-product of the research will be a solution to identifying necessary test conditions early in the design. Thus, the proposed research may produce results, which are widely applicable in various engineering and computer science areas.

#### 5. **Conclusions**

In addition to the above stated projects that have been directly impacted by the use of the equipment purchased through this NSF equipment grant, the equipment has also provided the necessary computational power for the following projects to students on campus working towards their Master's / Ph.D. thesis in engineering fields. Ms. Eik Lang Lau

from the chemical engineering program at Tennessee Technological University has used the systems for reaction modeling wherein she has used the machine for studying reactions between calcium hydroxide and fly ash using cellular automaton techniques.

#### 6. **Acknowledgements**

The above research projects were supported, in part, by a research equipment grant from the Division of Design and Manufacturing (DMI 9896235).

#### 7. **References**

1. Object Management Group, "Common Object Request Broker Architecture", <http://www.omg.org>
2. Jim Waldo, "The JINI Architecture for Network-Centric Computing", *Communications of the ACM*, July 1999. pp.77-82.