Secure Middleware for Situation-Aware Naval C^2 and Combat Systems*

In Proc. 9^{th} International Workshop on Future Trends of Distributed Computing Systems (FTDCS 2003)
May 28–30 2003. San Juan, Puerto Rico.

Dr. Ramesh Bharadwaj
Center for High Assurance Computer Systems
Naval Research Laboratory
Washington DC, 20375 USA
Email: ramesh@itd.nrl.navy.mil

Abstract

There is an increasing need within the Navy and Marine Corps for building distributed situation-aware applications that are rapidly reconfigurable and survivable in the face of attacks and changing mission needs. For the Navy's vision of Network Centric Warfare and the Total Ship Computing Environment to succeed, there is an urgent need for a secure, robust, and survivable network infrastructure for disseminating mission-critical information in a timely manner. It is widely believed that intelligent software agents provide the ability to build robust, agile, and efficient distributed applications. We outline how Secure Infrastructure for Networked Systems (SINS) being developed at the Naval Research Laboratory will provide commanders and warfighters the necessary middleware for constructing situation-aware Command and Control (C^2) and combat applications. We pay particular attention to the correctness, survivability, and efficiency of the underlying middleware architecture, and develop a middleware definition language Secure Operations Language (SOL) that enables C^2 and Combat applications to use this infrastructure in a seamless and scalable manner.

1 Introduction

Efforts are underway at the Department of Defense (DoD) for developing new technologies to create more effective sensor and communications architectures, enabling the Forces to create and exploit a common situational awareness and increase the speed of command and response. Termed Network Centric Warfare [6], this technology will provide warfighters with a new type of information advantage, broadly characterized by significantly improved capabilities for sharing and accessing information. A recent DoD report to Congress [Network Centric Warfare; Department of Defense Report to Congress, 27th July, 2001] identifies the following major technical and administrative impediments to progress in Network Centric Warfare:

- the lack of secure, robust connectivity and interoperability and
- the lack of technology investments in Network Centric Warfare.

Not only is robust connectivity important, but it is also imperative for the Information Network infrastructure to provide commanders with a situational awareness of their assets in the information battlespace and, in addition, to deny adversaries access to this information. For the vision of Network Centric Warfare to become a reality the DoD, including the Navy and Marine Corps, requires a network infrastructure for disseminating mission-critical information in a secure and timely manner. This is extremely difficult to achieve at present because Commercial Off The Shelf (COTS) products and legacy systems cannot provide fine-grained separation of classified data. Hence, this data is currently treated at the highest classification level. This leads to unnecessary downgrading of information-carrying data. Because current downgrading technology is unsophisticated and easily defeated by steganography and other clever coding schemes, the data is vulnerable to access by adversaries.

Another important requirement is rapid reconfigurability of the networked battle-space to satisfy the needs of new missions. This requirement is especially difficult to achieve in a coalition setting, where the need exists for interoperability between diverse systems and platforms and where the needs of the coalition

*This work is supported by the Office of Naval Research.

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
2003		**00-00-2003 to 00-00-2003**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Secure Middleware for Situation-Aware Naval C2 and Combat Systems	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Naval Research Laboratory,Center for High Assurance Computer Systems,4555 Overlook Avenue, SW,Washington,DC,20375**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**		**7**	

partners are in a constant flux. Also needed at the user level is programmability of the underlying application, such as a distributed mission planning system or a network centric Command and Control (C^2) for combat application, with a rapid turnaround time. As demonstrated by mission planning in recent Naval missions, current turnaround times are measured in days or weeks, rather than hours, and mission planning may involve hundreds of technical personnel working into the early morning hours. For Network Centric Warfare to succeed, the lengthy time and massive human resources needed to respond to new missions must be significantly decreased.

The ubiquity of the Internet and networking infrastructure, and the DoD's increasing reliance on computer networks for force coordination, mission planning, and mission definition, have created the need for a high assurance distributed computer platform which is secure, reconfigurable, and survivable. The benefits of Network Centric Warfare can be realized only after we address associated new security threats, both from malicious agents as well as untrusted or compromised hosts. The goals of Network Centric Warfare and the Total Ship Computing Environment[1], which are gaining prominence within the Navy and Marine Corps, will not be met if inadequate attention is paid to network and information security. Distributed computing will never gain wide acceptance if the associated middleware is unreliable and does not include capabilities to defend and protect vital information resources [7]. There is growing awareness within the defense research and development community that developing the next generation of sensor-rich, massively distributed autonomous systems will require a total paradigm shift in terms of the capabilities, processes, and architectures used to mitigate threats, plug vulnerabilities, and provide countermeasures.

The goal of the NRL Secure Agents project [5] is to develop a Secure Infrastructure for Networked Systems (SINS). It is widely acknowledged that intelligent software agents are central to the success of Network Centric Warfare. This is because agents are agile and provide an efficient and survivable paradigm for information distribution and access. Agents are efficient because only relevant information gets passed along. They are survivable because they are distributed. This new technology, which includes both autonomous and mobile agents, addresses many of the challenges posed by Network Centric Warfare. SINS is a middleware based on distributed agent technology. This middleware provides the required degree of trust in addition to meeting a set of achievable security requirements. Such an infrastructure is central to the successful deployment and transfer of agent technology to the Fleet because security is a necessary prerequisite for Network Centric Warfare.

The SINS project specifically addresses the following issues central to the effectiveness and security of a distributed agent architecture: (a) trustworthiness of the agents, (b) trustworthiness and timeliness of information gathered by the agents, (c) secure and timely propagation of information collected by the agents to the appropriate locations, (d) sharing of information from diverse sources, (e) sharing of information at different classification levels in a Multi Level Secure (MLS) environment, (f) more efficient, secure use of the limited Fleet communication resources, (g) collection of statistical data required to make correct tactical responses, (h) allowing tactical decision-making and responses from lower-level authorities, and (i) State-of-the-art visualization techniques and tools for the command center.

In a globally connected environment, computer-related attacks affect not only the host computer that is being attacked (or being used as a launch pad), but the network it is part of, not to mention the global infrastructure as a whole. Moreover, these attacks take only seconds or minutes to propagate and wreak havoc, unlike traditional tools of conventional or propaganda warfare that could take days or months to take effect. Therefore, current strategies in information warfare or Network Centric Warfare require fast detection of possible attacks, fast comprehension of the overall situation, and immediate and accurate responses and countermeasures to the situation. Tools built to support these strategies should also provide the flexibility and security services needed to ensure fast deployment, and secure communication between network hosts. The SINS infrastructure serves as enabling technology for network situational awareness. We address the infrastructure monitoring problem with the novel concept of security agents, which police the network, identify vulnerabilities, attacks, and compromised network components, and install effective countermeasures (such as rollback recovery, fail over recovery across domains, etc) to effectively deal with the problem.

2 Technical Approach

The goal of the SINS project at NRL is to develop enabling technologies and architectures to support a secure and reconfigurable infrastructure for networked C^2 for combat systems and network situational awareness. The results of this research will enable us to

[1] The computing environment for the DD(X) family of the next generation surface combatant ships of the US Navy.

build a network centric infrastructure for C^2 for combat applications to support the mission-critical needs of the Navy. Such an infrastructure is central to the successful deployment and transfer of network centric warfare technologies to the Fleet because a secure and flexible network infrastructure is a necessary prerequisite for Network Centric Warfare. To address the security issue, we have developed the novel approach of security agents [5] based on the notion of enforceable security policies of Fred Schneider [1, 11], to police the network infrastructure of distributed C^2 applications such as systems for distributed decision support and distributed mission planning. By using agent autonomy and mobility to our advantage, we ameliorate the security and safety vulnerabilities associated with agent technology. Security agents protect a network against Information Warfare (IW) attacks by including key security features such as encryption, authentication, virus checking, compliance checking, and intrusion detection. Security agents are therefore the enabling technology that give application developers the ability to deploy network centric systems which are secure and survivable, in a cost-effective and timely manner.

Since security agents have more privileges than secure agents, we have to provide assurance that their behavior will be safe. We have developed a special-purpose specification language Secure Operations Language (SOL) [3] to help provide this assurance. We are developing a SOL verifier (SOLver) to establish (with mathematical certainty) the compliance of security agents with their goals. We can also ensure that the behavior of security agents satisfies key security and safety properties [2]. The following technical issues [9] are being addressed in the SINS project:

- Consistency of security agent behavior
- Secure Operations Language (SOL)
  - How to make SOL agents composable, safe, and secure
  - Proofs that SOL security agents enforce required security policies
- Issues concerning Security Agents:
  - Authorization agents
  - Crypto assist agents
  - Policy enforcement agents
  - Secure agents monitoring
- Application-specific security agents:
  - Intrusion detection
  - Application monitoring
  - Survivability (adaptability)
  - Providing secure, safe, mobility of agent code
  - Making sure security agents enforce a consistent security policy
  - Network Situational Awareness and infrastructure monitoring
  - Developing a "Consistent Operational Picture" for Information Networks

The Secure Infrastructure for Networked Systems (SINS) and its associated Agent Creation Environment (ACE) are designed to explicitly solve the security problems described above and other related problems of agent creation and deployment. Security is our primary concern. However, while addressing security, we also intend to address problems of efficiency, reconfigurability, and survivability. Reconfigurability in ACE is supported by agent templates and other visual aids such as graphical visualization tools to ease the agent creation and customization processes. SINS provides role-based access control and management in addition to trust management. SINS will also include functions for intrusion detection and tolerance. SINS is designed for survivability and will support Multi-Level Secure (MLS) access and authentication. ACE supports visual Secure Operations Language (vSOL), a flexible and powerful notation in which to express the logic associated with an agent. For more details see [2, 3].

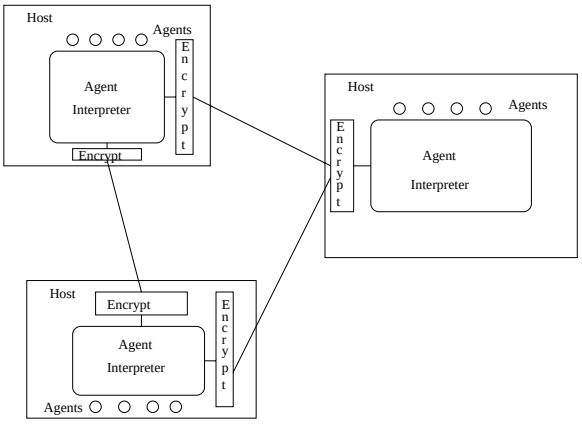

Figure 1. Architecture of SINS.

Figure 1 shows the architecture of SINS. Agents are distributed over one or more Hosts, each of which runs one or more Agent Interpreters (AI), that execute agents in compliance with a set of Security Policies. Agents are created using special-purpose templates in ACE (not shown), and are translated into SOL. Agents may be created on any host. Agent Interpreters com-

municate among themselves using a lightweight protocol similar to XML/SOAP [13], over secure channels, with strong encryption using a Public Key Infrastructure (PKI). For details of the SINS inter-agent protocol, see [12]. Our goal is to build a secure, survivable agent infrastructure that permits the Fleet to use commercial products safely. Hosts may run a COTS operating system such as Solaris or Windows, or a trusted operating system such as secure Linux (a product of the National Security Agency) or secure Solaris. We shall also investigate the use of other secure COTS components such as the secure Java Virtual Machine and other secure interpreters, as well as secure protocols that harness a public key infrastructure to distribute keys among interpreters and to authenticate agents.

2.1 Network Infrastructure for C^2 Systems

In a globally connected environment, computer-related attacks affect not only the host computer that is being attacked (or being used as a launch pad), but the network it is part of, not to mention the global infrastructure as a whole. Moreover, these attacks take only seconds or minutes to propagate and wreak havoc, unlike traditional tools of information warfare that could take days or months to take effect. Therefore, current strategies in information warfare or Network Centric Warfare require fast detection of possible attacks, fast comprehension of the overall situation, and immediate and accurate responses and countermeasures to the situation. These tools should also provide the flexibility and security services needed to ensure fast deployment, and secure communication between agents.

We address the following issues in SINS:

- Trustworthiness of agents: We provide formal arguments (proofs of correctness) for security agents that monitor the network infrastructure and implement the required security doctrine in case of an attack.

- Trustworthiness and timeliness of information gathered by agents: Information gathered by agents can come from various sources, requiring a method to identify friend vs foe, and a way to assess (weigh) the trustworthiness of the information. Also, to be effective, information should be collected and sent in a timely manner. The use of authentication techniques such as a Public Key Infrastructure (PKI) enable us to ensure agent trustworthiness.

- Secure and reliable propagation of information collected by agents: A temporally or spatially isolated view of attacks can give the wrong impression regarding an overall situation. Propagation of information regarding an attack on one target will enable heightened sensitivity at other potential targets, so that temporally and spatially distributed attacks may be successfully recognized and detected by a central authority. Furthermore, use of encryption ensures that while the information is propagated and shared among allies, it will be blocked from adversaries.

- Sharing of heterogeneous information (i.e., information from different sources): Information is gathered from various sources. For example, in Intrusion Detection Systems (IDS), attacks can occur at various levels, from physical components to the application level. Therefore, agents need to gather data from various sources such as log files, anomaly detectors, profiling data, intelligence sources, etc. We need a way to integrate this information from different sources for more accurate analysis and response. To render the information in a common format, we use XML to define common data formats, and rules and procedures for sharing information.

- Sharing of information among different classifications in an MLS environment: Situational awareness requires permissible information flow of need-to-know information only, between different security levels.

- Collecting necessary statistical data for appropriate strategic responses: To use agents successfully in intrusion detection, we need to apply distributed knowledge networks and data warehousing techniques to the infrastructure. These tools enable operations such as information retrieval, transformation, knowledge discovery, and data assimilation, on information from various heterogeneous distributed sources. This statistical data can also be used in developing battle space decision aids for information warfare purposes.

- Allowing tactical decision-making and responses from lower-level authorities: In a network centric environment, decisions may need to be made immediately. Our distributed infrastructure allows intermediate authorities to make tactical decisions regarding certain attacks or vulnerabilities without waiting for guidance/input from a central command (authority). For example, having received a CERT (Computer Emergency Response Team)

advisory of a new vulnerability, these intermediaries would be allowed to immediately exert remediation efforts, such as the installation of a patch or patches on computers under their administrative control. The intermediaries would also propagate this information to their chain of command, so that the central authority and other intermediaries could gain a quick understanding of the situation.

- State-of-the-art visualization tools for the command center: To fully cope with and comprehend the vast amount of information being transmitted and exchanged, we require a user-friendly visualization tool that displays the evolving state of the network. We are evaluating current visualization and image processing technologies to use them in developing unified and coherent visualization tools for this domain.

2.2 Network Situational Awareness

Although many Intrusion Detection Systems (IDS) have been developed commercially and are deployed operationally, their use has not decreased the number nor the severity of successful attacks on DoD systems. Information networks and systems are essential to network centric warfare and must be managed and controlled just like conventional forces. To assess the health of a network or to respond to an intrusion requires the correlation of incident reports from different types of IDSs that monitor different components in a network or system, and have different confidence levels. Fusing these disparate sources of incident data presents a daunting task for a network administrator. Additionally, sensors from different systems monitor different events of interest. Security alerts and intelligence reports must also be factored into these assessments. System and network managers alike, being overwhelmed with the amount of data and the diversity of incident reports, are unable to digest all the information in order to develop a reasonable response to an attack.

The Navy needs technology that can manage the propagation of intrusion detection information among many different organizational groups and to ensure that appropriate information is shared among all participating entities. This information will be processed and analyzed for different purposes throughout the enterprise. Additionally, certain groups have authority to direct responses to these attacks and to direct that security patches be made to Information Technology (IT) equipment. We need effective ways to carry out security patches and to respond rapidly to attacks based on the severity and magnitude of the attacks. In order to coordinate responses, decision makers need graphical representations of the current IT situation, which will require the correlation of large amounts of incident reports. Being able to include intelligence information in the decision making process would be very beneficial to rapid response and prevention of further damage.

In SINS, we address the problem of analyzing, filtering, coordinating, and communicating relevant incident data, to address the following requirements:

- Secure communication of incident reports and response strategies between organizations.
- Effective security alert approaches to ensure that security patches are applied in a timely manner.
- Confidence that the appropriate enterprise protection posture is maintained at all command echelons.
- Integration of intelligence data and reciprocal IW situational awareness sharing with the intelligence community.
- Assurance that the security management infrastructure itself is trustworthy.

3 Application of SINS to C^2 Systems

In this section we briefly examine how the network situational awareness of SINS is applied to combat applications. The application we shall use as an example is the Integrated Marine Multi-Agent Command and Control System (IMMACCS) [10], a Multi-Agent Decision-Support System prototype developed for the US Marine Corps. We begin with a brief introduction to the situation-aware middleware specification language Secure Operations Language (SOL) [3] and proceed to describe how functionality of IMMACCS may be implemented using our middleware.

3.1 A Brief Introduction to SOL

Agents are created in a special purpose synchronous programming language called Secure Operations Language (SOL) [2, 3, 5]. A SOL application comprises a set of agent modules, each of which runs on a given host. The host executes an agent module in compliance with a set of locally enforced security policies. A SOL multi-agent system may run on one or more hosts, spanning multiple networks and multiple administrative domains.

A module is the unit of specification in SOL and comprises variable declarations, assumptions and guarantees, and definitions. The `assumptions` section includes assumptions about the environment of the agent. Execution aborts when any of these assumptions are violated by the environment. The required safety properties of an agent are specified in the `guarantees` section. The `definitions` section specifies updates to internal and controlled variables.

A variable definition is either a one-state or a two-state definition. A one-state definition, of the form $x = expr$ (where $expr$ is an expression), defines the value of variable `x` in terms of the values of other variables in the same state. A two-state variable definition, of the form $x =$ `initially` $init$ `then` $expr$ (where $expr$ is a two-state expression), requires the initial value of `x` to equal expression $init$; the value of `x` in each subsequent state is determined in terms of the values of variables in that state as well as the previous state (specified using operator `PREV`). A conditional expression, consisting of a sequence of branches "`[]` guard $\rightarrow$ expression", is introduced by the keyword "`if`" and enclosed in braces ("`{`" and "`}`"). A guard is a boolean expression. The semantics of the conditional expression `if` $\{ \; [] g_1 \rightarrow \text{expr}_1 \;\; [] g_2 \rightarrow \text{expr}_2 \ldots \; \}$ is defined along the lines of Dijkstra's guarded commands [8] – in a given state, its value is equivalent to expression expr_i whose associated guard g_i is true. If more than one guard is true, the expression is nondeterministic. It is an error if none of the guards evaluates to `true`, and execution aborts. The case expression `case` $expr$ $\{ \; [] v_1 \rightarrow \text{expr}_1 \;\; [] v_2 \rightarrow \text{expr}_2 \ldots \; \}$ is equivalent to the conditional expression `if` $\{ \; [](expr == v_1) \rightarrow \text{expr}_1 \;\; [](expr == v_2) \rightarrow \text{expr}_2 \ldots \; \}$. The conditional expression and the case expression may optionally have an `otherwise` clause with the obvious meaning.

3.2 Issuing a Call For Fire

The Fires Agent of IMMACCS responds to "Call For Fire" (CFF) messages. The following are the logical rules associated with the functionality of issuing a CFF within IMMACCS. An agent may issue a CFF only if the forcecode is "not friendly" and the status of the locked-in radar is "ACTIVE". This requirement is captured by the following rule in the ACE front-end:

```
if Radar.forceCode == <friendly> &&
     Radar.status == ACTIVE
then
   CallForFire.target = name(Radar)
   CallForFire.controlMethod = WHEN READY
endif
```

```
deterministic module FiresAgent {
functions
  target_size = 20;
type definitions
  integer in [-20:100] ratings;
monitored variables
  integer CEP, ECR;
controlled variables
  ratings rating;
definitions
  rating = initially 100 then
  if {
    [] ECR < target_size -> PREV(rating) - 10
    [] CEP < ECR -> PREV(rating) - 5
    [] CEP > ECR -> PREV(rating) - 10
    otherwise -> PREV(rating)
  }
} // end module FiresAgent
```

Figure 2. A SOL agent to calculate the rating of a weapon.

3.3 Weapons Selection

The Fires Agent is also responsible for selecting the best weapon that is available, deliverable, and acceptable. The "rating" of a given weapon is based on the Circular Error of Probability (CEP), Effective Casualty Radius (ECR), availability, and Rules of Engagement (RoE). A (subset of) the requirements associated with this function is captured by the following ACE rules:

```
if Munitions.ECR < TargetSize
then rating = rating - 10
endif

if Munitions.CEP > Munitions.ECR
then rating = rating - 10
endif

if Munitions.CEP < Munitions.ECR
then rating = rating - 5
endif
```

The above rules are translated by ACE into SOL as shown in Figure 2. The formal semantics of SOL serves as the basis for analysis and transformation techniques for SOL specifications, such as abstraction, consistency checking, verification by model checking or theorem proving, and automatic synthesis of agent code [4]. For example, application of the tool SOLver on the above SOL specification will establish (with mathematical certainty) that it is free of ambiguity (i.e., it specifies exactly one action in any situation).

4 Conclusions and Operational Payoff

In this paper we show how SINS provides an integrated formal framework for the construction of situation-aware command and control applications. In particular, we examine the requirements of Network Situational Awareness for Naval C^2 and combat systems. The underlying formal framework of SINS serves as the basis for developing robust, efficient, and reconfigurable applications. Based on this framework, we are currently developing a suite of analysis and transformation tools for SOL, and verification tools such as automatic invariant generators and checkers, theorem provers, and model checkers. We currently have a compiler for SOL which generates Java code suitable for execution on multiple hosts. Planned extensions to the compiler include support for fine-grained access control and support for transactions, fault-tolerance, load balancing, and self-stabilization.

The SINS infrastructure provides a robust application development platform upon which networked C^2 for Combat Applications may be developed, tested, and fielded. SINS provides a seamless flow of information, with the desired quality of service, which is required to support not only horizontally distributed nodes but also vertical Command Echelons from the Commander-in-Chief (CINC) to the Unit level. The SINS infrastructure is fully end-user programmable and reconfigurable, with reconfiguration times measured in minutes instead of days or weeks. SINS will provide commanders and operators of networked C^2 systems the ability to request for and obtain the quality of service required to achieve the desired mission objectives. SINS is designed to be highly secure, having been built from the ground-up with quality control and high assurance in mind. Additionally, SINS is provably secure, i.e., free of flaws with mathematical certainty. Another important criterion we address in SINS is efficiency. Especially in a web-enabled and highly mobile setting, exchanging required information and only the required information saves bandwidth and reduces latency.

5 Acknowledgments

I thank Steven Yau for his invitation to participate in this workshop. I also thank Judy Froscher, Connie Heitmeyer, Amit Khashnobish, James Kirby, Unjung Ko Park, and John McDermott for many helpful discussions. I would also like to thank Connie Heitmeyer and the anonymous referees for their comments on previous drafts of this paper.

References

[1] B. Alpern and F. B. Schneider. Recognizing safety and liveness. Distributed Computing, 2:117–126, 1987.

[2] R. Bharadwaj. SINS: a middleware for autonomous agents and secure code mobility. In Proc. Second International Workshop on Security of Moble Multi-Agent Systems (SEMAS-02), Bologna, Italy, July 2002.

[3] R. Bharadwaj. SOL: A verifiable synchronous language for reactive systems. In Proc. Synchronous Languages, Applications and Programming, ETAPS 2002, Grenoble, France, Apr. 2002.

[4] R. Bharadwaj. A framework for the formal analysis of multi-agent systems. In Proc. Formal Approaches to Multi-Agent Systems, ETAPS 2003, Warsaw, Poland, Apr. 2003.

[5] R. Bharadwaj et al. An infrastructure for secure interoperability of agents. In Proc. Sixth World Multiconference on Systemics, Cybernetics, and Informatics, Orlando, Florida, July 2002.

[6] A. K. Cebrowski. Network Centric Warfare: Its origin and future. In Proc. Naval Institute, volume 124:1, pages 232–235, Huntsville, AL, Jan. 1998.

[7] A. Dey and G. Abowd. The Context-Toolkit: Aiding the development of context-aware applications. In Proc. Conference on Human Factors in Computing Systems, pages 434–441, ACM Press, USA, 1999.

[8] E. W. Dijkstra. A Discipline of Programming. Prentice-Hall, 1976.

[9] W. M. Farmer et al. Security for mobile agents: Issues and requirements. In Proc. National Information Systems Security Conference, Oct. 1996.

[10] J. Pohl et al. IMMACCS: A multi-agent decision-support system. Technical report, CAD Research Center, California Polytechnic State University, San Luis Obispo, California, Aug. 1999.

[11] F. B. Schneider. Enforceable security policies. ACM Transactions on Information and System Security, 3(1):30–50, Feb. 2000.

[12] E. Tressler. Inter-agent protocol for distributed SOL processing. Technical Report To Appear, Naval Research Laboratory, Washington, DC, 2002.

[13] W3C. Simple Object Access Protocol (SOAP) 1.1. Technical Report W3C Note 08, The World Wide Web Consortium, May 2000.

[14] S. Yau and F. Karim. Context-sensitive object request broker for ubiquitous computing environments. In Proc. 8^{th} IEEE Workshop on Future Trends of Distributed Computing Systems (FTDCS 2001), pages 34–40, 2001.

[15] S. Yau and F. Karim. Reconfigurable context-sensitive middleware for ADS applications in mobile ad-hoc network environments. In Proc. 5^{th} IEEE International Symposium on Autonomous Decentralized Systems (ISADS 2001), pages 319–326, Mar. 2001.