

Active Network Services for Mobile Wireless Environments

Ranga Ramanujan*, Ryan Hagelstrom, Maher Kaddoura, Sid Takkella, Ken Thurber, John Wu

Architecture Technology Corporation, 9971 Valley View Road, Eden Prairie, MN 55344

ABSTRACT

Active networks are a novel approach to network architecture that break from the traditional networking paradigm by allowing for the injection of customized computations into the network nodes. They provide a flexible network infrastructure that can be extended at runtime to implement the sophisticated network services needed to support the digitized battlespace of the future. This paper presents the design of an active network architecture that leverages the existing IP infrastructure to implement capabilities for the deployment of user-specified computations within an IP network. The Common Active Network Services (CANS) framework, described in this paper, allows for user-specified computations in the form of Java code to be injected into the network. Such computations are executed at strategic points within the network to implement new services targeted for mobile wireless applications.

Keywords: active networks, wireless networks, Internet Protocol, mobile computing, location-aware network services, ubiquitous networking

1. INTRODUCTION

Traditional packet switched networks passively move bits from one end of the network to another. The computations performed by the network nodes (routers and hosts) for processing packets is "hardwired" by the vendor. Furthermore, the role of computation within such networks is limited to header processing of the packets flowing through the network.

Active networks are a novel approach to network architecture that break from the traditional networking paradigm by allowing for the injection of customized computations into the network nodes. Thus, with active networking, the network is no longer viewed as a passive mover of bits, but rather as a programmable computation engine where information flowing through the network may be modified, stored, or redirected by customized functions as it is being transported. By providing a flexible network infrastructure that can be extended at runtime, active networks facilitate rapid deployment of the increasingly sophisticated network services required to support the digitized battlefield environment of tomorrow. In contrast, the evolution of traditional network services in the Internet environment follows a slow path of standardization, development, and deployment that typically takes 5-8 years.

Active networking opens up many exciting possibilities for implementing new network services to support future battlefield applications. The research effort reported in this paper designed an active network architecture that enables the implementation of new network services for mobile warfighters equipped with wireless computing devices (e.g., wearable computers, hand-held terminals). Examples of such new network services for the future battlefield environment include *ubiquitous networking services* and *location-aware wireless network services (LAWNS)*. Ubiquitous networking refers to the ability of mobile devices to access information and communicate with a worldwide network anytime and anywhere. This capability is critical to implement the network centric warfare strategy of tomorrow that relies on rapid dissemination of information across all levels of the warfighting force. Location aware wireless network services (LAWNS) exploits the widespread availability of low-cost GPS cards for mobile computers that make warfighter location information easily available. LAWNS uses GPS-based location information provided by warfighter equipment to implement new protocols to support capabilities such as geographically targeted multicasts of information, range-restricted information dissemination, and location-dependent access to distributed information services. Such a capability will enable future situational awareness systems where intelligence and tactical sensor data must be exploited to provide customized, location-specific information to the force.

* Correspondence: Email: ranga@atcorp.com; Telephone: (952) 829-5864

Our overall approach for implementing these wireless services relies on the strategic placement of several low-cost active nodes (or devices) within a fixed IP (Internet Protocol) infrastructure. These active nodes work in concert with legacy IP nodes (IP routers) to implement new services on top of IP. This approach preserves the existing investment in IP and applications running over IP while providing the capability to rapidly introduce new services. Section 2 describes the technical challenges associated with implementing two example wireless networking services, i.e., ubiquitous networking and LAWNS, in an IP network. It motivates the need for an active network architecture, such as that we propose, to implement such services. Section 3 present an overview of our active network approach for addressing these challenges. Section 4 describes the Common Active Network Services (CANS), the pivotal component of our active network architecture. Section 5 describes the results of a rapid prototyping effort that demonstrated the implementation feasibility of the active network architecture. Sections 6 and 7 present related work and conclusions, respectively.

2. CHALLENGES IN IMPLEMENTING NEW WIRELESS NETWORKING SERVICES

A key networking concept that enables ubiquitous connectivity for mobile wireless devices is that of a *wireless overlay network*¹. Such a network is formed by the unification of several heterogeneous wireless networks, of varying coverage and performance, into a single logical network that provides coverage that is the union of the coverage of the component networks with performance corresponding to the best network in range. Typically, in such networks, lower levels of the overlay consist of high-bandwidth wireless cells that cover a relatively small area. Higher levels in the hierarchy provide a lower bandwidth per unit area connection over a larger geographical area.

In such a heterogeneous environment, a mobile client equipped with the appropriate network interfaces for each of the wireless overlays can maintain connectivity with servers in the fixed network infrastructure as long as its roaming area is within the coverage of the highest overlay. Although the mobile client has ubiquitous connectivity to the server over the wireless network, the connection quality of the client may vary greatly by the location of the client. The connection bandwidth may vary, for instance, from megabits/second in a wired or in-room wireless LAN environment to a few kilobits/second in a metropolitan cellular network, to satellite networks with low bandwidth and high latencies but large coverage. In addition to these wide variations in network connectivity, the mobile clients themselves may have tremendous variations in their capabilities. Clients may vary along several dimensions, including processing power, screen size, color depth, the ability to handle specific data encoding (e.g., PostScript, GIF, PDF, JPEG, MPEG), and the ability to support network protocols such as IP multicast. High-volume devices such as PDAs, smart phones and smart two-way pagers will be increasingly used as clients in an IP network, making client variations even more pronounced.

The mobility of the clients, the variations in the clients' connectivity induced by their mobility, and the variations in the clients' capabilities present a number of challenges to the delivery of information services to these clients from the servers in the fixed infrastructure. Firstly, the information flow from a server must be robustly routed to a mobile client in spite of changes in its IP address or network interface induced by its roaming within and between the overlay wireless networks. Secondly, the heterogeneity of the clients and the dynamic variations in the clients' connectivity characteristics may require the information delivered to the client to be transformed in application-specific ways to fit the needs of the client. Furthermore, certain transformation may need to be performed dynamically, on-the-fly, to accommodate the changing connectivity characteristics of the clients as they roam. This makes it extremely difficult for servers to provide a level of service that is appropriate for every client. The active network architecture presented in this paper provides a solution to these problems.

The Location Aware Wireless Network Service (LAWNS) uses GPS-based location information to implement innovative protocols to enable the creation of location dependent services such as:

1. Multicasting selectively only to specific geographical regions defined by latitude and longitude, for example, sending orders or information to all warfighters who are currently in a specific area.
2. Providing a given service only to clients who are within a certain geographic range from the server (which may be mobile itself), say within 2 miles.
3. Advertising a given service in a range restricted way, say, within 2 miles from the server.

Implementing such services requires a routing protocol for geographic messages that would allow routing to a specific area defined, for instance, by a polygon of geographic coordinates. The main challenge therefore is to implement such a routing protocol over the Internet that currently relies on logical addressing (that is independent of physical location). The services provided by our active network architecture enable geographic addressing and routing to be implemented over IP networks without requiring any changes to the existing IP standards.

3. OVERVIEW OF ACTIVE NETWORK ARCHITECTURE

Our active network architecture for implementing new network services for wireless, mobile clients over IP networks is composed of three key elements that are described below, i.e., *surrogates*, *active nodes*, and *common active network services (CANS)*.

Surrogate: A surrogate is a Java-based mobile agent that resides on the fixed network infrastructure and performs application-specific computations (or functions) on behalf of a mobile client. The functions performed by a surrogate for a mobile client is tailored on demand for the mobile client when the mobile client seeks access to the network. The surrogate is then injected into the fixed network to serve as a smart intermediary that is interposed between the server and the mobile client. Figure 1 shows the basic surrogate architecture. As shown in the figure, by interposing itself between the server and the mobile client for all client-server interactions, the surrogate can effectively isolate the server from the vagaries of client mobility, client heterogeneity, and dynamic variations in the characteristics of a client's wireless connection. To accomplish this, the surrogate must implement application specific functions tailored to the needs of the client that it is serving. For instance, through various forms of data-specific transformations on the information flow from the server to the client, the surrogate can dynamically bridge the diverse needs of the clients and the formats and capabilities of the servers.

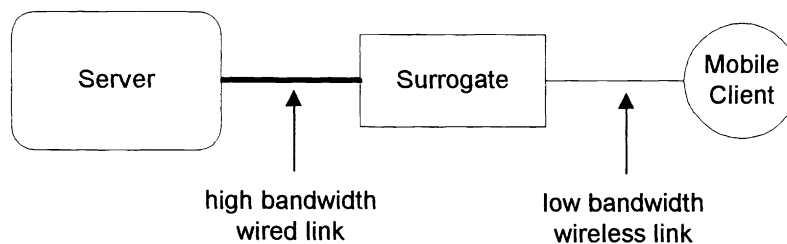


Figure 1: Role of a Surrogate in the Proposed Active Network Architecture

One example of data-specific transformations that can be implemented by surrogates is distillation^{1,2}, which is a highly lossy, real-time, datatype-specific compression that preserves most of the semantic content of a document or information stream (such as video). For example, the resolution of a color image can be scaled down and the number of colors reduced, thereby reducing the size of the image representation and making it more amenable to transfer over low-bandwidth wireless links. For video, distillation can be performed by reducing the frame size, frame rate, color depth, and resolution to create a smaller but reduced quality representation that can be accommodated within the constraints of a client's display capabilities and its network bandwidth. Intimate knowledge of each datatype allows distillation to achieve tremendous compression, sometimes better than a factor of 10, while preserving much of the semantic content of the original information².

Active Node: The computational resources that are needed to execute the surrogates within the fixed network infrastructure are provided by a pool of active nodes that may be dispersed over the fixed IP network. To support the execution of the Java-based mobile agents implementing surrogates, active nodes implement the Java Virtual Machine (JVM). Any commodity platform that provides a software or hardware implementation of the JVM can serve as an active node. Almost all commercially available workstations nowadays implement the JVM in software, including Windows and Unix platforms. In the near future, low-cost computing platforms providing a hardware implementation of the JVM (e.g., the picoJava architecture³) would become widely available. Therefore, active nodes in our architecture can be realized at low cost using off-the-shelf, commercially available technology.

Common Active Network Services (CANS): This is a key component of our active network architecture. It implements the active service framework to facilitate the injection of customized computations, i.e., surrogates, into the network at runtime. The major functions performed by CANS are:

Resource Monitoring: It monitors the status and utilization of all the active nodes in the pool to determine which active nodes are available to host new surrogates that may need to be instantiated to support new mobile clients.

Client Authentication and Access Control: It authenticates mobile clients that access the network to ensure that only authorized clients receive the computational resources (i.e., active nodes) for surrogates.

Client Profiling: It maintains a *profile database* containing the profiles of all the authorized mobile clients. The profile of a client describes all the applications must be supported by a wireless mobile client along with user

preferences associated with that client, the hardware configuration of the client, the wireless network interfaces supported by the client, and so on.

Software Repository: It serves as a repository of Java-based software components that implement various application-specific functions, such as the distillation functions for various data information types.

Surrogate Customization: It configures client-specific surrogates for authorized mobile clients.

Resource Control: It controls the allocation of surrogates to active nodes in the network.

Figure 2 presents a layered view of the structure of the active network architecture that was described in the paragraphs above. The common active network services (CANS) layer implements those active network functions that support the implementation of all other wireless network services. Thus, new services such as ubiquitous networking services and LAWNS use the facilities of the CANS layer to implement the application-specific network computations needed to support these services. The applications layer builds on the ubiquitous networking services and LAWNS layer to implement various wireless networking applications for the mobile clients.

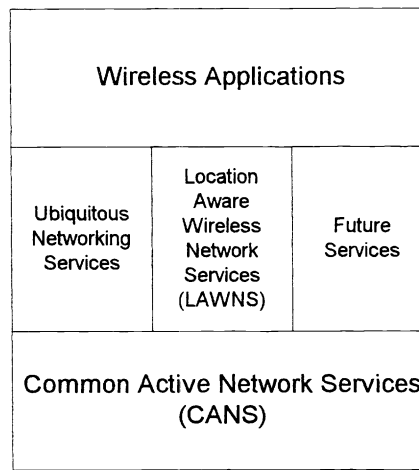


Figure 2: Layered Structure of the Active Network Architecture

4. COMMON ACTIVE NETWORK SERVICES (CANS)

The common active network services (CANS) framework provides a secure and fault-tolerant environment for mobile clients to customize and inject computations into the fixed network infrastructure. This approach moves computational and storage complexity away from the mobile devices into the fixed infrastructure, thereby enabling the cost effective implementation of new services for a variety of small, lightweight, low-power, inexpensive mobile devices such as PDAs, smart phones, and smart pagers.

The key abstractions and terminology for the architecture of CANS are illustrated in Figure 3. Mobile wireless nodes that require CANS services are called *mobile clients*. The service management and control (SMAC) server implements the CANS functionality described in Section 3, i.e., resource monitoring, client authentication and control, client profiling, software repository maintenance, surrogate customization, and resource control. SMAC servers may be replicated for fault-tolerance. Each fault-tolerant SMAC server group is responsible for servicing the set of mobile clients assigned to its *domain*. A mobile client may belong to one or more domains and can only receive services from a SMAC server in its domain. Active nodes form a pool of available computational engines, distributed over the Internet, that host surrogates launched by the SMAC service in response to requests from mobile clients.

The operation of CANS can be summarized by the following steps that are performed when a mobile client requests services:

1. The mobile client sends a service request to the SMAC server.
2. The SMAC server authenticates the client and determines the admissibility of the service request.
3. Using the software components maintained in its software repository, the SMAC server configures (assembles) a tailored surrogate for the client.

4. The SMAC transfers the assembled Java bytecode for the surrogate to an available active node in the network.
5. The selected active node loads the Java bytecode and starts the execution of the surrogate.
6. The newly instantiated surrogate opens a communication session with the mobile client and starts providing its services to the client.

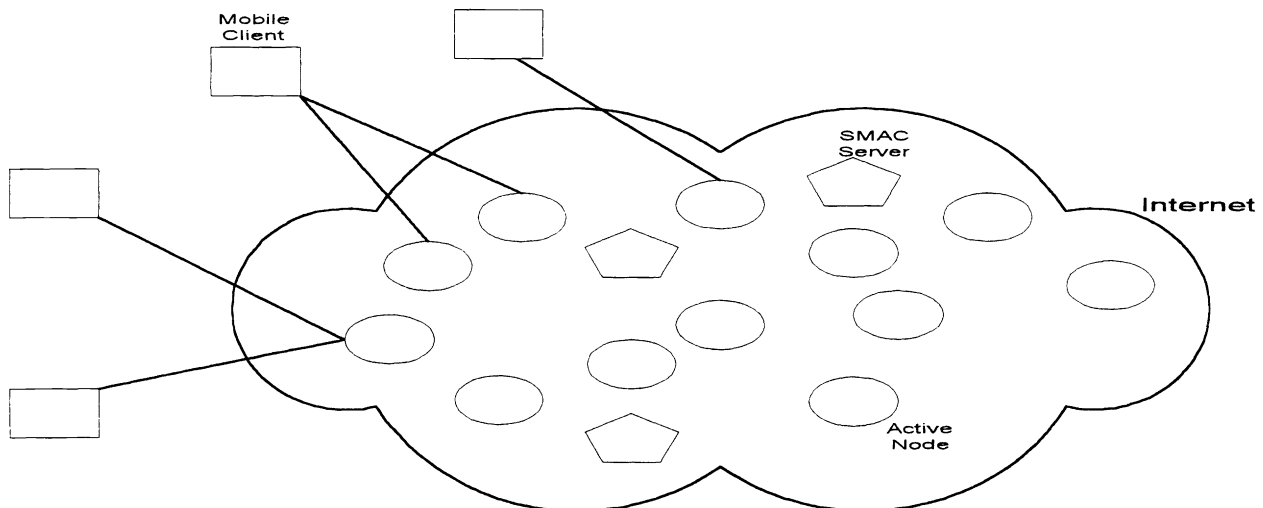


Figure 3: Architectural Elements of the CANS Framework

The SMAC server is designed to be a trusted system that is maintained by the administration authority for the active network. All active nodes are maintained, administered, and managed by the SMAC server. The SMAC server is the only entity that can inject software into the active nodes. The network administration authority has total control over the mobile Java code maintained by the SMAC servers. Thus, the surrogate code can be assumed to be trusted. CANS implements mechanisms that allow an active node to authenticate and verify the credentials of a SMAC server. An active node will only accept mobile code from authorized SMAC servers. Similarly, CANS implements mechanisms that allow SMAC servers to verify the credentials of an active node before transferring surrogate code to the active node. Surrogates only execute on active nodes trusted by the SMAC server. Taken together, these mechanisms provide a secure execution environment for the surrogates. A mobile client is ensured that its surrogate executes on a trusted platform within the network whose security can be guaranteed by the trusted network authority.

A mobile client maintains a list of unicast IP addresses of SMAC servers that it is subscribed to. When the mobile client needs the services of CANS, it rendezvous with one of the SMAC servers in this list. Should this fail, it attempts to connect with another SMAC server in the list. Thus, even if some of the SMAC servers in the client's list were unavailable, the mobile client would be able to access the services of CANS.

All active nodes under the control of a SMAC server can be reached via a multicast IP address. The SMAC server and the active nodes in the multicast group maintain a shared secret key that can be used to encrypt message exchanges among the members of the group.

The following paragraphs describe in more detail the message exchange protocol used between the mobile client and the SMAC server and between the SMAC server and the active nodes to implement CANS functions.

The mobile client connects to the SMAC server. The latter sends a nonce (random number) to the client. The mobile client sends a service request to the SMAC server including the nonce and digitally signs it with its private key. The service request contains two fields: 1) a service request ID (SRID) that uniquely identifies this service request from all previous and future service request from the client; and 2) a service descriptor that specifies the active network service requested by the client. The service descriptor includes information to identify the code for the surrogate needed to implement the requested service.

When the SMAC server receives the service request from the client, it uses the client's public key maintained in its database to verify the digital signature within the message and thereby authenticate the client. The use of the nonce precludes replay

attacks from malicious nodes masquerading as a legitimate client. Once authenticated, the SMAC server examines the service request to identify the code in its software repository that is needed to implement the surrogate for the client.

To select an active node to host the surrogate, the SMAC server multicasts a Resource Request to the pool of active nodes. The multicast message is encrypted using the shared group key for the multicast group. The Resource Request includes a digital certificate issued by a well-known certification authority that contains the identification of the SMAC server (i.e., its IP address) as well as the SMAC server's public key. The message is signed by the SMAC server using its private key.

When an active node receives a Resource Request, it first verifies the identity of the sender of the message using the information in the message. If the message originated from an SMAC server that it trusts and if the active node is available to host a surrogate, it may send a Resource Response to the SMAC Server. Similar to the Resource Request message, the Resource Response message carries information that can be used by the SMAC Server to authenticate the sender of the message.

To prevent the SMAC server from being overwhelmed by a flood of Resource Response messages arriving from a large number of responding active nodes, CANS employs a damping mechanism for its resource management function which operates as follows. An active node does not immediately send its Resource Response message after processing a Resource Request. Instead, it sets a random timer called the *response timer*. The active node waits for the response timer to expire before sending a Resource Response. When a SMAC server receives a Resource Response, it verifies the authenticity of the sender. If this succeeds, it multicasts a Cancel Timer message to the multicast group containing the SRID of the Resource Request sent by it previously. On receiving a Cancel Timer message, active nodes that had response timers set for the SRID in the message cancel their timers.

The SMAC server loads the Java code for the surrogate into the selected active node. It then provides the mobile client with the information necessary for it to rendezvous with the surrogate. Thereafter, the client and the surrogate communicate using an application-specific protocol.

5. PROTOTYPING RESULTS

To demonstrate the implementation feasibility of the active network architecture described above, we built and experimented with a rapid prototype of the system. The rapid prototype implemented a slice of the CANS layer of the active network architecture. This CANS prototype provides the rudimentary capabilities of a programmable network substrate on which one could build arbitrary network services for wireless environments. Each new network service is defined by the set of surrogates associated with the service.

For the experimenting with the CANS prototype, we implemented two simple wireless network services, i.e., a stock quote service for wireless handheld devices (e.g., Palm VII) and a stock monitoring service for mobile wireless clients (e.g., smart pagers).

The stock quote service is targeted for wireless thin clients, such as the Palm VII, that are severely constrained in terms of their computational power, display area, and the bandwidth of their wireless connection to the network. The Palm VII, for instance, has a display area of 160X160 pixels and an 8 Kbps wireless link. The Web pages delivered by popular stock quote services (e.g., NASDAQ, Stockmaster) cannot be displayed in their original formats on such devices. Furthermore, we found that the volume of data associated with these Web pages typically varies anywhere from 25 Kbytes to 100 Kbytes. Transporting this volume of traffic for a Web page has two major implications for the user. First, for some wireless environments such as that for the Palm VII, the user is charged for the wireless services based on the volume of data transported over the wireless network. Currently, for the Palm VII, this cost is \$0.30 per kilobyte of data. Thus, it becomes prohibitively expensive for Palm VII users to browse through such large Web pages. Second, the low-speed wireless link greatly increases the Web access latency for such heavyweight Web pages.

The stock quote service is implemented using a surrogate that is interposed between the stock quote server and the mobile client and that performs semantics-based transformation or distillation of Web pages delivered by the server. Our experiments with the rapid prototype of this network service indicate that the distillation produces two orders of magnitude reduction in the volume of data that is sent over the wireless network. Figure 4 depicts the average traffic volume reduction delivered by the stock quote service as a function of the number of stock symbols in the quote request made by the mobile client.

Currently, the Palm VII facilitates efficient access to selected Web content via “Web clipping” services. However, this requires content providers to implement this service that is specifically tailored for Palm VII users. Furthermore, the users’ choice of Web content is limited to those sites that implement this service. Our active network approach described above overcomes this limitation by empowering clients to inject surrogates that can perform user-customized, semantics-based transformation of Web pages.

The second wireless network service, i.e., the stock monitoring service, is implemented by a surrogate that periodically monitors the set of stocks specified by the client to determine whether any of them have crossed their client-specified price thresholds. A threshold crossing results in a notification message being sent by the surrogate to the mobile client over the wireless network. The surrogate in this case can be viewed as implementing a “smart-push” type of a network service for the wireless environment thereby minimizing cost and bandwidth consumption on the wireless network.

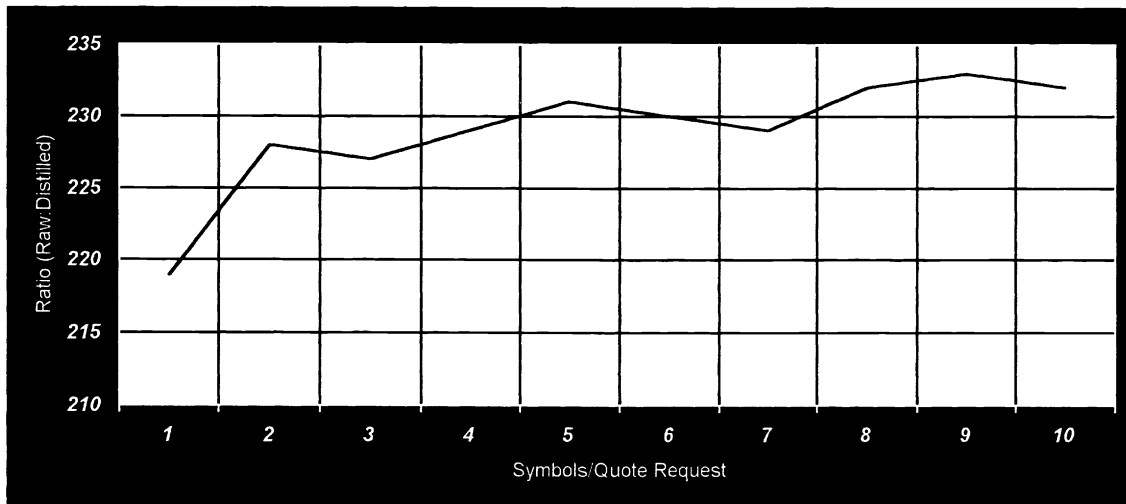


Figure 4: Average Traffic Volume Reductions Produced by Semantics-Based Transformation (Distillation) for the Stock Quote Service

6. RELATED RESEARCH

Tennenhouse et al^{4,5} defined a general model for the implementation of active networks. This implementation model contains two primary elements:

Capsules: These are generalized replacements for packets in traditional networks. They carry an executable program specifying processing to be performed on their behalf by the active nodes through which they pass.

Active nodes: These are replacements for routers and end nodes in traditional networks that execute capsules and maintain state. They export a set of primitives for use by application-defined processing routines associated with capsules. These routines are composed by combining the node primitives using the control constructs of a programming language.

The general active network model described above forms the basis of several alternative active network implementations including ANTS⁶, CANES⁷, and Switchware⁸. These alternative implementations primarily differ in the flexibility with which capsule routines can be composed using the primitives of the active node. ANTS allows capsule routines to be composed using the control constructs a full-fledged, object-oriented language, i.e., Java. Switchware provides a programming language called PLAN with much restricted constructs. In CANES, capsule routines are specified as a simple ordered sequence of node primitives to be invoked by the capsule.

The primary goal of all the active network architectures described above is to provide a general mechanism that would allow new network services to be deployed at any layer above layer 2 of the 7-layer OSI protocol stack model. However, this generality comes at a price. The packet forwarding performance of active routers implemented by these projects does not approach anywhere near the performance of traditional IP routers. Techniques for improving the performance of active

routers are still being investigated. In addition to performance, another formidable problem associated with a general model of active networks is security. This is another area where solutions are being investigated^{9,10,11}.

Our approach for implementing an active network for wireless services adopts a much limited model of active networks. Our model is targeted at the deployment of network services on top of IP, at the application layer. While this model lacks the generality of the implementations mentioned above, it is sufficiently rich to allow a wide range of wireless services to be implemented. Furthermore, the performance and security issues in this approach are much more easily addressed than in the general model. Finally, our approach allows for the leveraging of the existing IP infrastructure and commercially available computing platforms to implement new and useful services in the immediate future.

The idea of performing application-level function on nodes within the network to implement wireless networking services has been explored recently in the context of proxy architectures^{1,2}. The proxy approach represents an ad hoc approach for migrating application functions within the network. Incremental insertion of new functions within the network is difficult with such an approach. Our active network approach, on the other hand, provides a structured way of injecting application functions into the network that greatly simplifies incremental refinements and extensions to these functions as well the addition of new functions. More importantly, unlike the proxy approach, our active network architecture allows for client-specific run-time tailoring of application functions. This allows for a much richer and useful set of network services to be implemented than is possible with the proxy approach. Finally, our approach uses Java-based mobile code for implementing the application specific functions within network nodes. This allows for a wide variety of low-cost commercial computing platforms for executing these functions. These platforms can be dispersed in any fashion across the IP network thereby facilitating cost-effective scaling of the processing capacity of the active network. In contrast, the proxy architecture relies on platform-dependent proxies that must execute on a local-area network of workstations configured as a server farm. Emir et al¹² present an active service framework, called AS1, for deploying application-level computing functions within an IP network. However, AS1 lacks the secure environment provided by CANS for the execution of surrogates. Furthermore, unlike CANS, AS1 requires clients to periodically transmit service requests. This is problematic in mobile wireless environments with power constrained thin clients or in battlefield environments with low probability of intercept (LPI)/low probability of detection (LPD) requirements.

In terms of related work in the area of location-based network services, Imielinski and Navas¹³ proposed three alternative schemes for implementing GPS-based routing. However, unlike LAWNS, all these schemes require changes to the network infrastructure (e.g., changes to IP's unicast and multicast routing protocols and changes to DNS).

7. CONCLUSIONS

Active networking presents many exciting possibilities for implementing the sophisticated network services needed to support the digitized battlespace of the future. This paper presented the design of an active network architecture that leverages the existing IP infrastructure to implement capabilities for the deployment of user-specified computations within an IP network. The Common Active Network Services (CANS) framework allows for user-specified computations in the form of Java code to be injected into the network. Such computations are executed at strategic points within the network to implement new services targeted for mobile wireless applications.

The research effort reported here successfully implemented a rapid prototype of the critical elements of the Common Active Network Services (CANS). It also implemented and demonstrated two example wireless applications using the services provided by CANS. The encouraging results of this rapid prototyping effort served to establish the implementation feasibility of our active network architecture for wireless environments.

Our ongoing research work seeks to extend the functionality of CANS to implement two additional capabilities: 1) support for sharing surrogates among multiple mobile clients to facilitate a more efficient implementation of shared network services; and 2) resource allocation strategies for optimal placement of surrogates within the network.

ACNOWLEDGEMENTS

The research effort reported here was supported by the Office of Naval Research (ONR) under the Small Business Innovation Research (SBIR) program.

REFERENCES

1. E. Brewer, R. Katz, et al, "A Network Architecture for Heterogeneous Mobile Computing," IEEE Personal Communications, Oct. 1998.
2. A. Fox, S. Gribble, Y. Chawate, and E. Brewer, "Adapting to Network and Client Variation Using Infrastructural Proxies: Lessons and Perspectives," IEEE Personal Communications, Aug. 1998.
3. H. McGhan and M. O'Connor, "PicoJava: A Direct Execution Engine for Java Bytecode," IEEE Computer, Oct. 1998.
4. D. Tennenhouse and D. Wetherall, "Towards an Active Network Architecture," ACM Computer Communication Review, Apr. 1996.
5. D. Tennenhouse, et al., "A Survey of Active Network Research," IEEE Communications Magazine, Jan. 1997.
6. D.J. Weatherall, J.V. Guttag, and D.L. Tennenhouse, "ANTS: A Toolkit for Building and Dynamically Deploying Network Protocols," IEEE OPENARCH'98, San Francisco, CA, Apr. 1998.
7. S.Bhattacharjee, K.L. Calvert, and E.W. Zegura, "An Architecture for Active Networking," Proceedings of High Performance Networking 97, June 1997.
8. J. Smith, "Switchware: Accelerating Network Evolution," Tech. Report MS-CIS-96-38, CIS Dept., Univ. of Pennsylvania, 1996.
9. D.S. Alexander, W.A. Arbaugh, A.D. Keromytis, and J.M. Smith, "A Secure Active Network Environment Architecture: Realization in Switchware," IEEE Network Magazine, May/Jun. 1998.
10. Special Issue on Active and Programmable Networks, IEEE Network, May/Jun. 1998.
11. Special Issue on Programmable Networks, IEEE Communications Magazine, Oct. 1998.
12. E. Emir, S. McCanne, and Randy Katz, "An Active Service Framework and its Application to Real-Time Multimedia Transcoding," Proceedings of ACM SIGCOMM'98, October 1998.
13. T. Imielinski and J. Navas, GPS-Based Addressing and Routing, IETF RFC 2009, Experimental Category, Nov. 1996.