Active IP Network Node Developments

A. Tan <atan@ee.ucl.ac.ul>, A. Galis <a.galis@ee.ucl.ac.uk>
Department of Electrical & Electronic Engineering
University College London

Abstract: *This paper presents the on-going active node development effort in the IST project called Future Active IP Networks (FAIN). It begins by introducing the active networking paradigm, and underlines the motivation behind the project. A new concept of active node is presented using a flexible, and generic architecture. It concludes with some open issues and challenges to be addressed in the project.*

1. Introduction to Active Networks

In today's Internet, routers examine the destination address field of the IP header along with the internal routing tables to determine to which neighbour they should forward the IP packet. The extent of user control over the network's behaviour is thus limited to the range of values that can be placed in that field in the IP header. The network consists of smart hosts sitting at the edges of the network that are capable of performing computations up to the applications layer [Pso99], and routers (simply packet-forwarding engines) that interconnect the hosts and can only perform computations up to the network layer.

An active network allows intermediate routers to perform computations under the end-user's control up to the application layer (which means that routers can perform computations on user data). This effectively exposes the underlying resources, mechanisms, and policies of the router to the end-user. Active networks explore the idea of allowing routing elements to be extensively programmed by the packets passing through them. The raison d'être for the strong interest in active networks is their potential for accelerating network evolution. With active networks, once the service is defined, the network can be customised 'on the fly' [Ale97] for the delivery of that service, without delays due to lengthy standardisation procedures.

The network application-programming interface (APIs) defines a virtual machine that interprets a specific language for the Internet Protocol (IP). It comprises the language defined by the syntax and semantics of the IP header and its effects on the routers on the network [Cal98]. In traditional networks the virtual machine is fixed, and the expressive power of the language is limited. Active networks can be seen as providing 'programmable' network API. If the IP header is seen as the input data to a virtual machine, packets in active networks contain programs as well as input data.

2. The FAIN Project and Motivation

In response to critical demands to achieve suitable trade-offs among flexibility, high performance, security and manageability for active network nodes, the FAIN project is aimed at investigating the integration of distributed object technology, mobile agent technology and active networking technology, to find a flexible and secure solution. The need to provide novel node architecture to establish demarcation between various actors and roles in the business and enterprise model for telecommunications has also inspired the direction of active node development in the project. The following section chronicles a new concept of active node by means of a flexible, and generic architecture. This is followed by a description of the active node platform, followed by its service execution environment. This paper concludes with some open issues and challenges to be addressed in the project.

2.1 FAIN Node Architecture

The Active Network Node (ANN) forms the backbone of the active network technology. A three-tier new node architecture is envisaged, as depicted in Figure 1 below [Gal00]. It represents a generic framework for developing the elements in an active network. An active network thus consists of a set of such active nodes connected by a variety of network technologies, e.g., ATM, Ethernet, and UMTS.

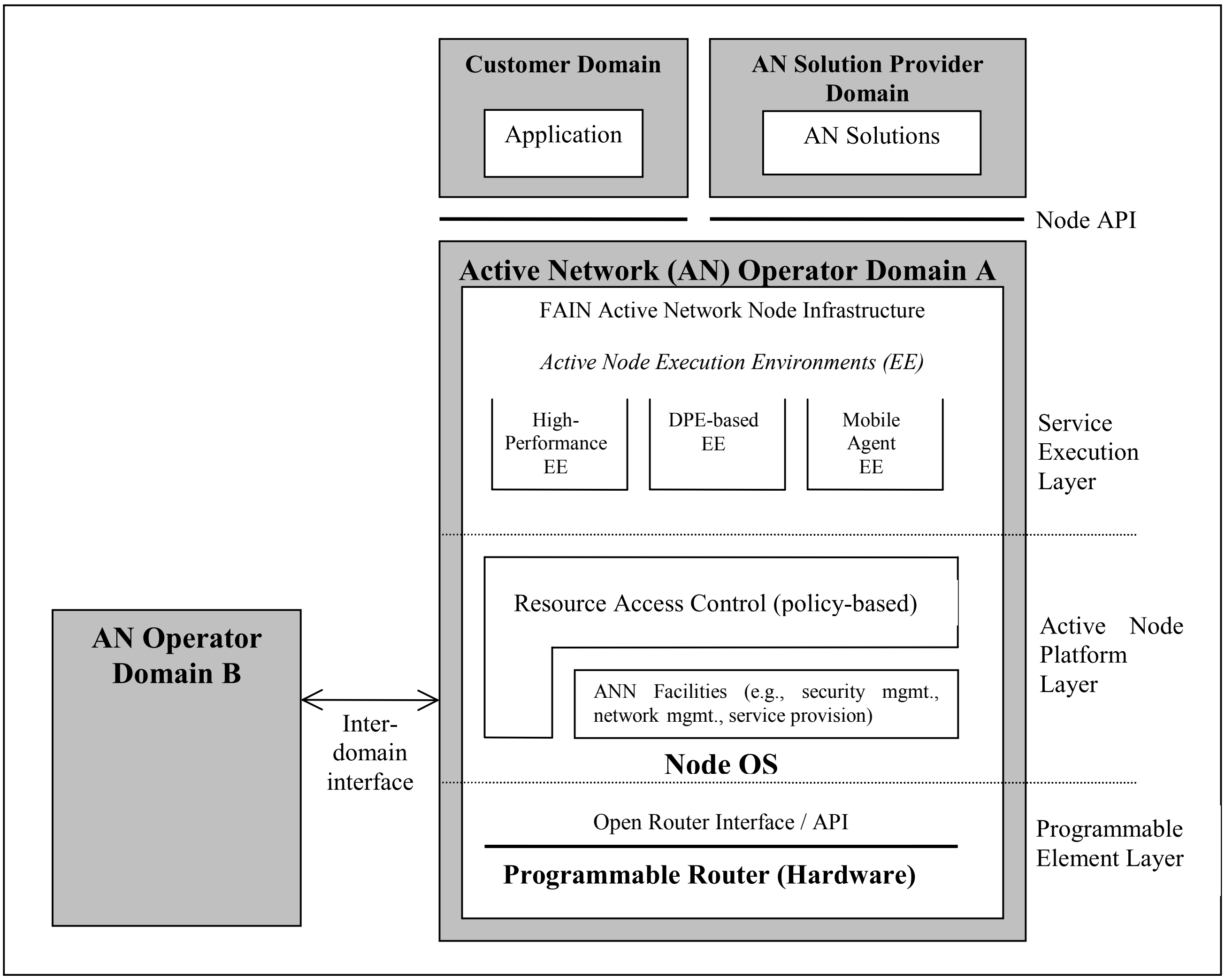

Figure 1: Initial FAIN Active Node Architecture

To have an open and secure control of network resources, an active node is built upon one programmable network element, e.g., an IP router with an open interface. The computing platform in the node provides a layer through which downloaded/injected components interact with networks and with each other. In general, it consists of a local operating system; one or more distributed processing environment, and system facilities. Upon the platform, a set of execution environments (EEs) will be created to host service components.

Additionally, an open node interface will be defined, and it represents an abstraction of router resources ranging from computational resources (e.g., CPU, and memory) to packet forwarding resources (e.g., bandwidth, and buffer). The specification of the open AN node interface in FAIN will serve as an input to relevant standardisation activities, e.g., IEEE P1520 or IETF.

2.2 Active Node Platform

The node platform can be viewed as a container of ANN services, which represent the local node services and provide the foundation for the execution of service components, which are usually in a network-wide scope. To guarantee a secure and fair use of resources, the platform defines a resource control framework that partitions and allocates resources (including computing resources such as CPU time and memory, and network resources such as bandwidth and routing table). The framework implements the API as an abstraction of the partitioned resources, which will be used by EEs.

The resource framework and the active network facilities will be designed as the services of a distributed processing environment (DPE). FAIN proposes to allow implementations of these services in different DPEs, depending on specific requirements in terms of performance, or functionality. An active node is under control of a node operating system (NodeOS). Node platform provides the basic functions on which EEs rely. The NodeOS mediates the demand for resources, including transmission, computing, and storage among EEs and isolates them from each other.

A single node is expected to support multiple concurrent EEs. Different EEs may have different trust levels; and the NodeOS will protect the node by enforcing boundary and resource limits on each of its EEs. The EEs in turn hide most of the details of the platform from the users and implement the network API. An active node allows existence of a number of environments for execution of software components. These environments can be built around different DPEs, and thus it is very likely that heterogeneous components co-exist in a single active node. Distribution of the components into environments could be implemented using the active network encapsulation protocol (ANEP), which provides the capability for users to have their packets routed to a particular EE at a node [Bra97].

The five core abstractions defined by the NodeOS–EE interface are thread pools, memory pools, channels, files and flows [Pet00]. The first four encapsulate a system's four types of resources, which are computation, memory, communication, and persistent storage. The fifth abstraction, the flow, is used to aggregate control and scheduling of the other four abstractions.

2.3 Service Execution Environment

An EE can be created by or on behalf of the consumer or AN solution providers, to meet application-specific requirements. Such an EE supplies a restricted form of service programming, in which the user supplies a set of parameters and/or simple policy specifications. In any case, the program may be carried via the integrated programmable switch approach or the discrete capsule approach [Ten97].

The programmable switch approach maintains the existing packet/cell format, and provides separate mechanism that supports the downloading of programs (i.e., programs and data carried separately). Taking it a step further with the capsule approach, active miniature programs that are encapsulated in transmission frames replace the passive packets of present day architecture. Here, every message is a program.

In FAIN, the issue of incompatibility between these two approaches will not be further researched since it has been proven in ALIEN [Ale99] that the distinction between the programmable switch (active extension) and capsule (active packet) models is a distraction rather than a central issue in active networks.

3. Conclusion

The FAIN active network architecture will support new business models of network control and management and a wide range of distributed applications as envisioned in the future information society. Dynamic customisation of multiple EEs is an important issue to address. Such a capability enables maximal customisability by users, but raises serious concerns of security. Spawning these application-specific EEs could be done by a bootstrap EE owned by trusted authority, i.e. active network operators. One key novel capability envisaged in the FAIN project is the existence of EEs that runs the middleware components of dedicated DPE and mobile agents.

The last facet of active networks is its management system, which is responsible for controlling the behaviour of all network nodes under the manager's jurisdiction. The nodes must securely receive and process active packets, and the management system must ensure this happens. A crucial issue to be addressed is the demonstration of a resilient network management framework, whereby QoS, resource management, security, and integrity issues are resolved. Otherwise, service providers and network operators will not be susceptible towards the active networking approach to provide value-added services. There is also a challenge to exhibit large-scale implementation, with the interaction of components in different environment an open issue for the project to investigate. A comprehensive solution for interoperability of multi-domain active networks is envisioned and the validation exercise will involve Pan-European trials with international links to USA.

4. References

[Pso99] K. Psounis, "Active Networks: Applications, Security, Safety, and Architectures", IEEE Communication Surveys 1999

[Ale97] D. S. Alexander, M. Shaw, S. M. Nettles, and J. M. Smith, "Active Bridging", In Proceedings, SIGCOMM Conference, 1997

[Cal98] K. L. Calvert, S. Bhattacharjee, E. Zegura, and J. Sterbenz, "Directions in Active Networks", IEEE Communications Magazine, Vol. 36, No. 10, October 1998

[Gal00] A. Galis, B. Plattner, J. M. Smith, S. Denazis, H. Guo, C. Klein, J. Serrat, G. T. Karetsos, C. Todd, "A Flexible IP Active Networks Architecture", IWAN 2000

[Bra97] D. S. Alexander, B. Braden, C. A. Gunter, A. W. Jackson, A. D. Keromytis, G. J. Minden, D. Wetherall, "Active Network Encapsulation Protocol (ANEP)", Draft RFC, Active Networks Group, July 1997

[Pet00] L. Peterson, ed., "NodeOS Interface Specification", AN Node OS Working Group, 24 January, 2000

[Ten97] D. Tennenhouse, J. Smith, W. Sincoskie, D. Wetherall, G. Minden, "A Survey of Active Network Research", IEEE Communications Magazine, pp. 80-86, January 1997

[Ale99] D. S. Alexander, J. M. Smith, "The Architecture of ALIEN in Active Networks", IWAN '99, June 1999