A Flexible IP Active Networks Architecture

Alex Galis[1], Bernhard Plattner[2], Jonathan M. Smith[3], Spyros Denazis[4], Eckhard Moeller[5], Hui Guo[5], Cornel Klein[6], Joan Serrat[7], Jan Laarhuis[8], George T. Karetsos[9] and Chris Todd [1]

[1] University College London, Department of Electrical and Electronic Engineering, Torrington Place, London WC1E 7JE, United Kingdom
{a.galis, c.todd}@ee.ucl.ac.uk
[2] The Swiss Federal Institute of Technology, ETH Zurich, Computer Engineering and Networks Laboratory ETH-Zentrum, CH-8092 Zurich, Switzerland
plattner@tik.ee.ethz.ch
[3] University of Pennsylvania, Department of CIS, Philadelphia, PA 19104, USA
jms@central.cis.upenn.edu
[4] Hitachi Europe Ltd., Information and Telecommunication Laboratories, 59 St Andrews Street, Cambridge CB2 3BZ, United Kingdom
sdena@hitachi-eu.com
[5] GMD- FOKUS, Kaiserin-Augusta-Allee 31, D-10589 Berlin, Germany
{moeller, guo}@fokus.gmd.de
[6] Siemens ICN M RP 11, Hofmannstr. 51, D-81359 München, Germany
Cornel.Klein@icn.siemens.de
[7] KPN Research, Dept. Communications Architectures and open Systems, P.O. Box 15000, 9700 AC Groningen The Netherlands
J.H.Laarhuis@research.kpn.com
[8] Universitat Politècnica de Catalunya, Dept Teoria del Senyal i Comunicacions, Jordi Girona, 1-3, 08034 Barcelona, Spain
serrat@tsc.upc.es
[9] National Technical University of Athens, Telecommunications Laboratory, Division of Electrical Engineering, 15773 Zographou, Athens, Greece
karetsos@softlab.ece.ntua.gr

Abstract. This paper presents the main concepts of the IST Project FAIN "Future Active IP Networks" [10], a three-year collaborative research project, whose main task is to develop and validate an open, flexible, programmable and dependable network architecture based on a novel active node approach. This generic architecture for active networks is an innovative integration of active networking, distributed object and mobile agent technology. Starting from the definition of a business model that underlines the FAIN architecture, we identify three key working areas for contribution: the active node platform layer, the service programming environment and a built-in management system.

The active node platform layer of the FAIN Active Node is comprised of the kernel Operating System, a node resource access control framework, and active components for management, security and service provision. These elements provide the foundations for Execution Environments so that they can operate in an independent manner. A novel service programming environment is envisaged to enable the dynamic creation or update and to secure deployment and operation of protocols. Such an environment supports various role-specific ways of deployment, e.g. application-specific signalling or operator-governed network control signalling.

1 Introduction

The wide acceptance of IP has enabled the provision of *new application services*. The popularity of IP originates from its unparalleled ability to provide ubiquitous access and low prices regardless of underlying networking technology. These services can be offered on a global scale by almost everyone, simply by connecting a new web server to the Internet. Today IP [20] is considered a unique bridge for diverse application/user requirements with broadband transfer capability.

However the development and deployment of new network services, i.e. services that operate on the IP layer, is too slow through best practice and standardisation. It cannot match the rapid growth of requirements in various applications. Examples of such services include signalling for quality of service (QoS), reliable multicast or Web Proxies/Caches/Switches/Filters. As with the intelligent network (IN) architecture in the PSTN world, the current Internet architecture needs to be enhanced in order to allow for a more rapid introduction and programmability of such services.

The Internet community has realised the need for network-embedded functionality, and has been addressing these needs on a problem-centric basis rather than on an architectural basis. For example, various approaches to differentiated -service architectures have applied the idea of active queue management in routers, resulting in algorithms such as RED and FRED, etc. which provide a reasonable class of service performance. As a second example, the MBONE architecture identifies MBONE flows, which are segregated from "regular" traffic by participating routers, among which, traffic is "tunnelled".

Active Networks (AN) have been originally proposed [25] as an architectural solution for the fast and flexible deployment of new network services. The basic idea of active networks is to enable third parties (end users, operators, and service providers) to inject application-specific services (in the form of code) into the networks. Applications are thus able to utilise these services to obtain required support in terms of network and network management resources, thus becoming network-aware. As such, active networks allow *dynamic injection of* code for realisation of-application-specific service logic, or perform dynamic service provision on demand. But the dynamic injection of code can only be acceptable to network providers if it does not compromise the integrity, the performance and /or the security of networks. Therefore viable architectures for active networks have to be carefully engineered to achieve suitable trade-offs among flexibility, performance, security and manageability.

Programmability, a fundamental requirement of active networks, means that a user is able to control dynamically the network to process packets in a required way, rather than in a fixed fashion, e.g., by selecting/implementing preferred routing algorithms. Active networks normally implement code distribution and code execution mechanisms to enable such ability, so that injecting code programmed by users can enforce control of the networks. Another facet of programmability is that network services (developed as active network applications) use the open router interface (implemented as an API) to dynamically manage router resources.

The IEEE P1520 [4], [9] standardisation effort addresses the need for a set of standard software interfaces for programming of networks in terms of rapid service creation and

open signalling. The technology under consideration spans from ATM switches and IP routers, to circuit or hybrid switches. Well-defined open interfaces represent abstractions of resources of the underlying physical network devices and are implemented as distributed computing objects. These open interfaces allow service providers and network operators to manipulate the states of the network through the use of middleware toolkits in order to construct and manage new network services.

The active networks research has made it clear that software-implemented active network elements and simple applications such as active pings and TCP through active bridges can perform in the 10-100 Mbps range. ANTS [26] and PLAN [21] have shown that capsules are in fact a viable concept, and in the case of PLAN, that minimal-function capsules can perform reasonably. The SwitchWare [16], [23] has demonstrated that mutually distrustful nodes can support remote-module loading using cryptographic hashes. ALIEN [2] has demonstrated 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. An approach for designing high performance active network was demonstrated in [8].

What has yet to be designed is an integrated hardware/software system, which performs to a high level, is secure, flexible, programmable, manageable and usable by a variety of applications. Moreover, it provides the basis on which a number of Execution Environments of different technology and goals will be deployed and tested against each other. These EEs may eventually take the form of different network architectures resulting in proper VPNs.

The FAIN project [10] has been set-up to demonstrate such a system, and address key problems of resource partition and control, security, dynamic loading of protocols, and dynamic management in a new active network node environment. FAIN is a 3 year IST collaborative research project among universities, research establishments, manufacturers and operators starting in May 2000. The project aims to develop an open, flexible, programmable and dependable network architecture based on novel active node concepts. The generic architecture for active networks is an innovative integration of active networking, distributed object and mobile agent technology. The project will contribute to the establishment and operation of a worldwide active networks and services test bed.

The remainder of the paper is organised as follows: Section 2 discusses the design of the FAIN Active Networks Architecture. Section 3 suggests the enterprise model on which FAIN is based. Section 4 addresses the node architecture. Section 5 addresses the management of the active networks. Section 6 identifies the FAIN testbed. Section 7 concludes and outlines challenges for the future.

2 The FAIN Design Approach

In defining FAIN architecture, our goal was a system, which would allow experimentation and prototyping to test Active Networks ideas. From the architectural viewpoint, FAIN defines a network architecture, which is based on the combination of traditional layer-based networking and distributed component-based networking. Such a

combination of architecture design brings many advantages, for example, modularity and location transparency for service provisioning, which facilitates fast changes and smooth integration of new services and network technologies.

Our design follows a bottom up approach, originating from the design of the AN platform layer and its components and moving towards the service programming environments held together by the built-in management system with the following characteristics.

The active node platform layer of the FAIN Active Node is comprised of the kernel OS, a node resource access control framework, and active components for management, security and service provision. These elements provide the foundations on which Execution Environments are deployed and operate in an independent manner.

A novel service programming environment enables the dynamic creation or update of protocols, and supports various role-specific ways of deployment, e.g. application-specific signalling or PNO-governed network control signalling. This environment is secure and maintains the interference-free execution semantics of different protocols or components, so that safe execution of protocols can be guaranteed and the network behaviour is predictable.

In addition, FAIN proposes a management paradigm based on standardised API and autonomy of nodes. It is based on the approach identified in [11] and [12]. The paradigm enables the development of a fine-grained and more efficient management framework, which reduces needless traffic or information processing. Examples include filtering and self-management of nodes, which take care of the management of their own resources and states. Autonomous management of nodes enables the distribution of management intelligence. Loosely coupled management functions facilitate the traditionally difficult tasks such as policy enforcement and integration of new managing functions. Re-usable components and interoperable operation can be achieved using the standard interface and an implementation using distributed objects and platforms.

The FAIN architecture is based on a new enterprise model. In this way the FAIN architecture supports new business opportunities:

- The project enables services to be delivered to Active Network Operators as products that can be installed and executed according to service level agreements (SLA).
- The project also identifies two new business players: active middleware providers, and active network solution providers.

3 Enterprise and Network Models

Open telecommunications solutions will involve a variety of roles and players on the value chains with heterogeneous requirements. The FAIN project is based on an enterprise model of the European Information Infrastructure [22] with some enhancements, as dictated by an initial requirement analysis conducted by related active networks projects [16]. The enterprise model is depicted in the Figure 1. The definition of FAIN enterprise model specifies the business relations among various players in future active networks.

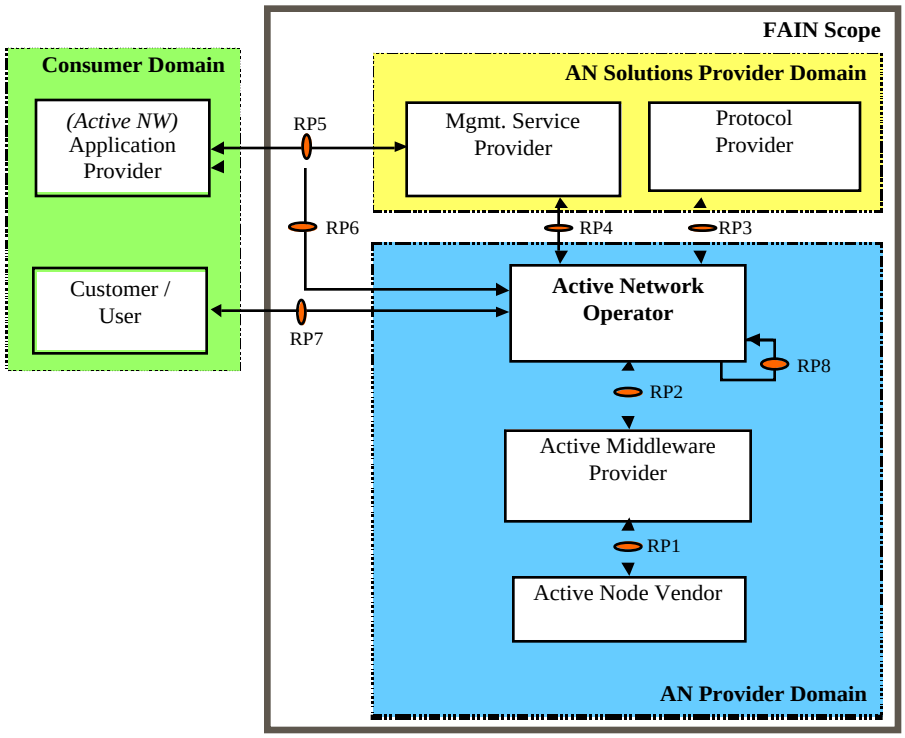

Figure 1. Initial FAIN Enterprise Model.

FAIN active network architecture

Traditional network architectures are usually tied to the interface presented by the network to the user (e.g. in the form of UNI, NNI). An architecture for active networks typically addresses the interface between AN providers and users, i.e. consumers and solution providers, and should be explicitly designed to allow for more than one such “network API". The following figure depicts the initial network architecture envisaged in the FAIN project. In addition the project will address the requirements of key functions in the provision of an active network:

- *Service Provision:* The objective of this activity is to identify the architectural requirements for flexible support of a range of services, including application services and network services. Focus will be put on service execution and management environment. Mechanisms for code distribution and code execution will be a major requirement.
- *Security Provision:* The objective here is to define the architectural requirements for security provision in active networks. It focuses on three principal aspects:

(1) Trust-based software distribution: only trusted parties are allowed to download their components into networks for special provisioning of services. This is realised by the

operations of authorisation and authentication of user/code/node in active networks. The authorisation could be based on a predefined service level agreement (SLA).
(2) Trust-based software execution: only trusted components (based on a component level agreement) are allowed to access and manipulate a particular set of network resources (e.g. in the form of virtual networks) and sensitive information resources.
(3) Policy-controlled resource access, and runtime resource management should support this interference-free software execution: software components run independently for provision of services, and their interaction will be policed and controlled to ensure that an abnormal execution of one component will not negatively impact on other components' execution. This should be supported by independent execution environments (EE), a guarantee of integrity of the shared data, and transaction-based interaction among components.

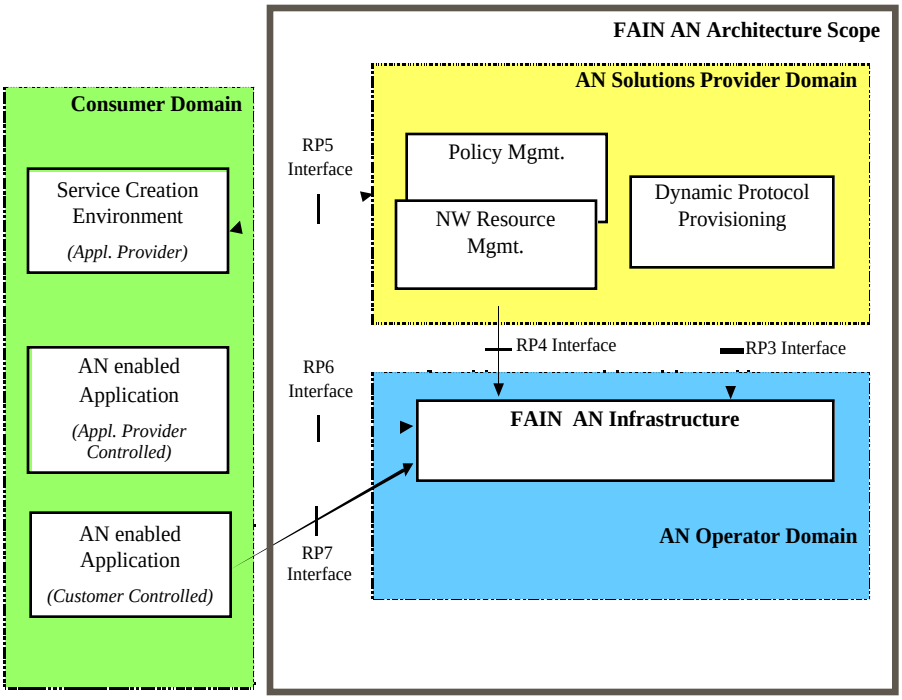

Figure 2. Initial FAIN Active Network Architecture

Management Service: The objective here is to specify the requirements for managing the active nodes, node resources and services. The focus will be on supporting service provisioning, including resource management functions (partitioning and policing); service management, including service installation, upgrade and supervision; network management functions as configuration and monitoring of nodes and links.
Network Integration: The objective of this part is to specify the system requirements for integrating the functions for service provision, security, and management within

an active node and over a unified active network. Of typical focus are interoperability and dependability (mainly reusability and reliability) of active nodes.

4 FAIN Nodes and Networks

Central to the active network technology is the active network node. In FAIN, the design of the AN Nodes will be based on the specification of FAIN Enterprise Model. We envisage a three-tier node architecture as depicted in Figure 3. It represents a generic framework for developing the elements in an active network. An active network thus consists of a set of active nodes plus –the possible addition of traditional ("passive") nodes, connected by a variety of network technologies, e.g. IP, IP/ATM, Ethernet, UMTS, etc.

An active node is an essential network element in the support development of value-added solutions by third parties, or direct use by consumers (users or application developers) to inject customised services. It implements APIs according to the requirements of reference points (R3, R4, R6 and R7 in Figure 3) defined in the business model. 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 (e.g. Linux, free BSD or an embedded OS), one or more distributed processing environment (e.g. TINA-DPE, or mobile agents platform) and system facilities.

Upon the platform, a set of execution environments (EE) will be created to host service components. Solution providers according to security agreements can download such components. They can also be dynamically injected by consumers to implement a service for particular applications. Their execution is controlled so that services they support run safely in parallel, and the behaviour of the node and the network is predictable.

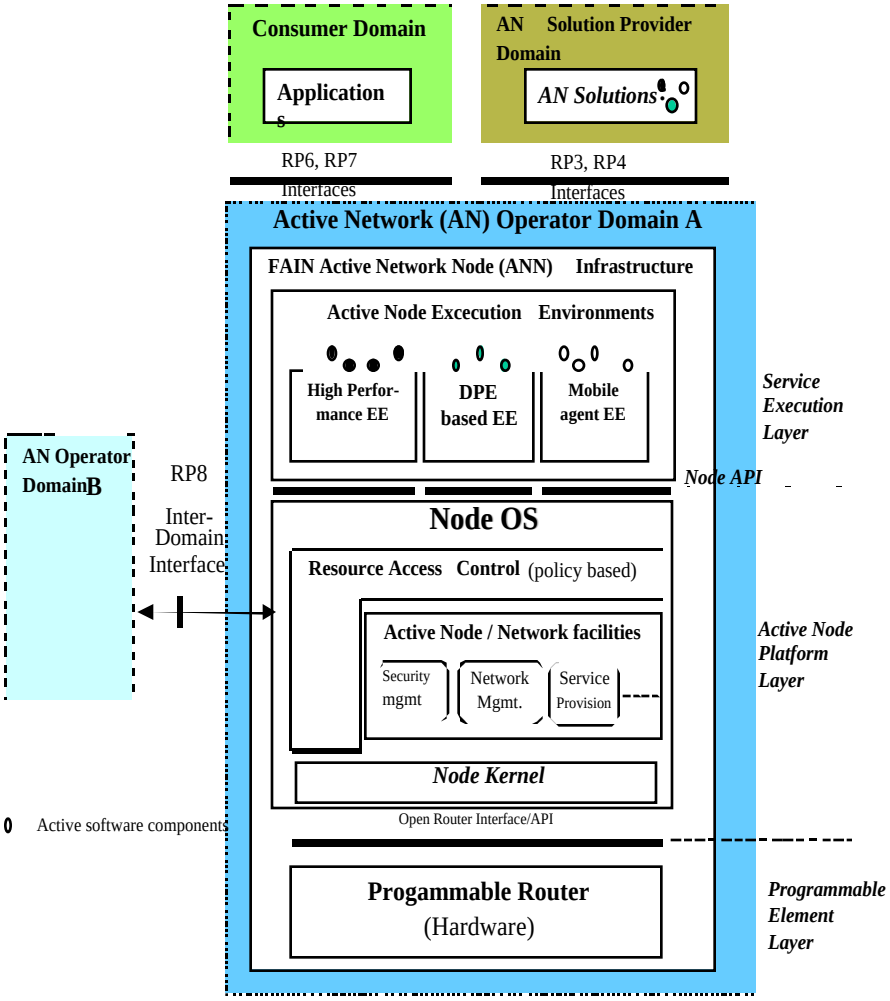

Figure 3. Initial FAIN Active Node Architecture

Programmable network element

Active networks are built upon a programmable network infrastructure that mainly comprises of programmable routers and the open interface for controlling the routers. Router hardware will be contributed by router vendors while their OS will be enhanced to account for the identified AN requirements. Among potential enhancements are advanced IP services, e.g. Diffserv, for selective package processing, partitioning resources such as virtual network to support VPN, and native mechanisms for identifying active packets from incoming data flows, and distributing them to appropriate execution environments.

In addition, an open node interface that actually represents an abstraction of router resources ranging from computational resources (CPU, memory, etc.) to packet forwarding resources (bandwidth, buffer, etc.) will be defined. The specification of the open AN node interface may also serve as an input to relevant standardisation activities, e.g. IEEEP1520 [3] or IETF [13] and other consortia [18].

Active Node Platform

The node platform can be viewed as a container of AN node services, called hereafter facilities. These facilities represent the local node services and provide the foundation for the execution of service components, which are usually in a network-wide scope. Example facilities are:

- *Binding facilities* that support flexible and dynamic QoS binding of distributed components;
- *Management facilities* that provide transparent access of resources (e.g. a MIB), and enforce policies when necessary;
- *Service facilities* that support distribution (downloading or injection) of components (as CORBA [6] objects, mobile agents [11] or program segments carried by packets);
- *Security facilities* that perform trust-based authentication of software components when distributed, and runtime state checking to ensure safe execution.
- *Communication facilities* that provide the service components with connectivity using flexible protocol bindings.

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 an execution environment. It also implements a policing entity that enables policy-based management, i.e. enforcing and checking the access to node resources.

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. DPE could be based on TINA-DPE [24], real-time ORB, JAVA virtual machine [27], mobile agent platform [14], or other distributed platforms. As a major contribution, an AN DPE will integrate a dedicated ORB and a mobile agent platform to provide a full range of signalling services and real-time QoS control for distributed applications. In order to support the needs of distributed multi-media and real-time bound applications, a specific execution environment will be provided which is optimised for high performance, i.e. high packet processing rates and low latency. At last, active network services can be provided to satisfy the very diverse requirements of applications in the future information society.

Node platform provides the basic functions on which execution environments rely. As such, it takes the form of an OS, manages the resources of the active node and mediates the demand for resources, including transmission, computing, and storage. It thus isolates EEs from the details of resource management and from the effects of the be-

haviour of other EEs. The EEs in turn hide most of the details of the platform from the users and implement the Network API.
AN technology have advocated the co-existence of few Execution Environments (EE) at each node where each one of them implements a different virtual machine [5] on top of the NodeOS [19]. To the best of our knowledge, no project has implemented and integrated such architecture in a large-scale deployment. The only similar effort we are aware of is the Tempest Framework [17] for ATM networks.

Service Execution Environment
Supported by the node platform, an active node allows existence of a variant number of environments for execution of software components. These environments can be built around different DPEs, and so, it is very likely that heterogeneous components co-exist in a single active node. The interoperation of components running in the same-type environment is guaranteed by the underlying DPE (e.g., by IIOP). The interaction of components in different environments will be an open issue to investigate in the project, towards a full solution for interoperability of active networks. Distribution of the components into environments relies on the underlying active network facilities.
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 in-band with the packet itself, or installed out-of-band. Out-of-band programming may occur in advance of packet transmission or on-demand, upon packet arrival automatically (e.g. when instructions carried in a packet invoke a method not present at the node, but known to exist elsewhere), or under explicit user control. These EEs are customisable via interfaces visible to the trusted users of the EE. Dynamic customisation of these EEs is an important issue to address. Such a capability on one hand enables maximal customisability by users, but on the other, raises serious concerns for 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 run the middleware components of dedicated TINA-DPE and Mobile Agents. These middleware components would facilitate further expansion of the scope and flexibility for the mobility and programmability in the Active Networks.
Generic EEs will be developed to support particular types of services. An EE specialised for management services will host the components conveying management functions, e.g. node configuration. It will define an interface accessible by node managers to customise the environment parameters so that a policy can be enforced dynamically. Generic management functions will be developed in the management EE as EE-inherent components to support application management requirements.
Another generic EE foreseen as critical for provisioning active networks is one for protocols/signalling that perform network control functions, e.g. packet scheduling. Such an environment should smoothly support dynamic provision of protocol components, and their update. The result is that deployment and new signalling can be

safely executed. Multi-media applications and real-time bound applications need high-performance active network services, which currently cannot be deployed in a full-featured ORB-based execution environment. For such applications, a high-speed execution environment is provided. It will be based on the highly efficient Node OS, augmented with a thin layer of resource access functions as a minimum.

Service programming
In an active node, new services can be introduced as distributed components. These components will be downloaded or injected using distribution mechanisms in the node platform and executed in execution environments. One such component is allowed to interact with others and the services provided by the platform, e.g. resource manager and facilities through Node API. The results of the interaction lead to the execution of the services and changes of the node state. These components can be remotely controlled or dynamically programmed, and thus the services they implement can be flexibly customised.
As envisioned, building applications either as value-added solutions or consumer-specific services are both based on this service programming paradigm, and aligned with the interface specification identified in the FAIN Node architecture.
The architecture proposed in the project is generic and supports specialisation according to different needs of service provision, security and management. For example, one Active Network Operator (ANO) can only implement the service environments into an embedded OS, which controls the data path and provides expected performance when re-routing the packets. The safe execution is guaranteed by trust-based download, i.e. only it is allowed to download the service components.

5 FAIN Network Management and Applications

Based on the network API, two different end-to-end Active Network solutions will be designed and developed, demonstrating the potential capabilities of FAIN system. The two Case Studies will be:

- Policy-based Active Network Management, which will provide flexible network configuration, efficient network error detection and dynamic resource management through continuous interworking among active nodes.
- Dynamic Creation of Protocols, which will enable the rapid provision and update of protocol stacks.

The objective of developing these two services on top of the FAIN architecture is to demonstrate the benefits of active networks in general and to evaluate the applicability of the FAIN architecture in particular. To demonstrate the applicability of the FAIN architecture with respect to the provisioning of end-to-end network services, two case studies will be designed and implemented: Policy-Based Network Management and a demonstration of the dynamic creation of protocols.

Policy-based Active Network Management

Policy-Based Networking (PBN) is currently promoted by several network equipment vendors (e.g. DTMF) and is standardised within the IETF Policy working group. The current goal of PBN is to provide facilities, which allow control the multiple types of devices that must work in concert across even a single domain. Examples of services, which can be controlled by PBN, are currently mainly Quality of Services (QoS) reservations. Examples of devices, which can be "PBN-enabled", are hosts (clients and servers), routers, switches, firewalls, bandwidth brokers, sublet bandwidth managers and network access servers. Policies are defined as rules governing the allocation of network resources to certain applications/users. The IETF Policy WG defines a scalable framework for policy administration and distribution that will allow interoperability among the multiple devices that must work together to achieve a consistent implementation of a network administrators policy. For this reason, directory schemas are standardised in order to enable the various network devices to interpret the configuration information consistently. For the distribution of policy information, IETF protocols such as LDAP, DIAMETER, COPS are used.

In FAIN, service logic for the PBN-based provisioning of a sample service (e.g. virtual private networks or QoS policies, depending on progress in the IETF) will be implemented on top of the Node API. With this case study, we expect to be able to demonstrate the following benefits of active networks:

- *Flexibility:* Using active networks, policies can not only be expressed as data structures - which are limited by the constraints imposed by standardised database schemas, but they can be expressed quite flexibly as *active code.*
- *Simplified service deployment:* Since all the FAIN nodes support the same Node API, we expect that the same PBN service logic may run on the different nodes. In contrast to current approaches, where the service logic has to be developed individually by each node vendor, this results in an improvement of the service development and deployment process with respect to costs and time to market.
- *Benchmarking:* Since PBN-enabled network devices are expected to be available on the market soon, the PBN case study allows benchmarking with existing PBN implementations, in particular with respect to performance, interoperability and reliability.

Dynamic Creation of Protocols

In a second case study we will demonstrate how application-specific code can be injected into active networks in order to provide application-specific network services. A specific application/service has still to be chosen for that purpose. Several applications which come into consideration for that purpose have already been mentioned (e.g. QoS signalling, Web Caching/Web Content Adaptation/Web Switching, reliable multicast...).

While in the PBN case study, an *existing* service will have been prototyped with FAIN, this second case study will demonstrate a distinct feature of Active Networks, namely that they allow the rapid introduction of *new network protocols* by third parties or even by skilled end-users. Properties of the architecture which are important for that purpose and which will be demonstrated include:

- interference-free execution of the different protocol stacks (i.e. a malicious protocol stack does not affect the correct operation of other protocol stacks);
- security issues;
- monitoring and logging of service execution, in particular with respect to resource consumption.

6 FAIN Testbed

In the FAIN project a networking test bed will be built. It has at its core a number of sites running an Active Network Node, which is designed and implemented within the project (Figure 4). The individual sites are connected through IP tunnels over the Internet (possibly with guaranteed quality of service), leased lines or native ATM links. The main purpose of this test bed is to assess the results of the project using experimental scenarios and quantitative measurements.

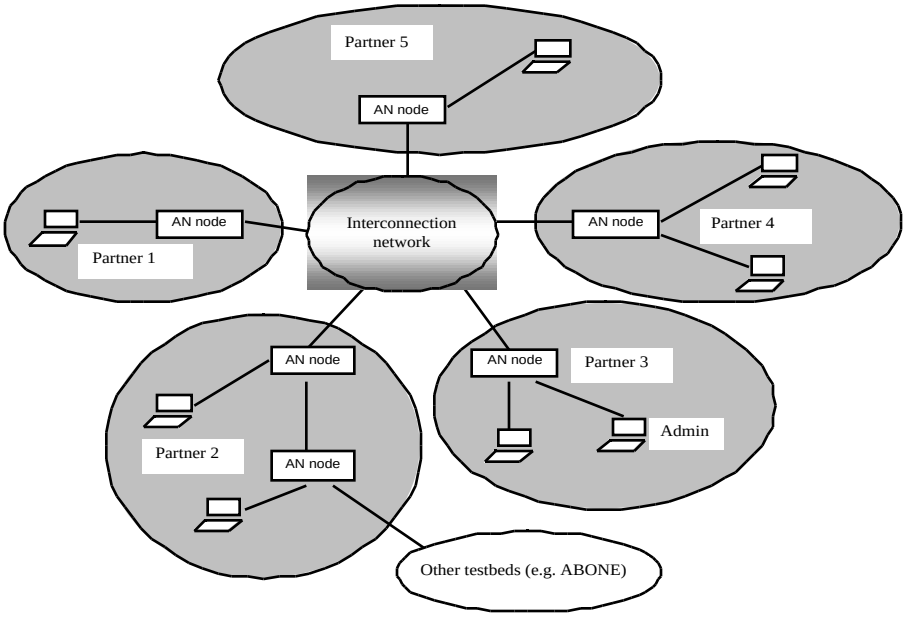

Figure 4. Active Network Test Bed

By running different active network services on the test bed, it will be shown that the interworking of different active network implementations from various partners is possible, thereby providing a worldwide test bed. The different implementations will be built according to the specifications developed on the FAIN Node Architecture and will support multiple execution environments with different levels of performance. The testbed will serve to test the degree of programmability and interworking issues among ANN with different execution environments, while interoperability tests will be carried out among ANNs that have common execution environments.

The envisaged test bed is depicted in Figure 4. The active network consists of 4 - 8 sites at partner locations, with one or more ANNs and several terminal devices. ATM links or IP-tunnels interconnect the sites. The terminal devices access ANNs through network technology available, including Ethernet, Fast-Ethernet, and ATM. Therefore FAIN will form the one of the first ever worldwide Active Network test beds interworking with other testbeds. Management service provision and a dynamic protocol provision case studies are envisaged for the demonstration of the FAIN test bed.

7 Conclusion

This paper gives an overview of the IST Project FAIN, a three-year project, whose main task is to develop and validate an open, flexible, programmable and dependable network architecture based on novel active node concepts. The generic architecture for active networks is an innovative integration of active networking, distributed object and mobile agent technology. The project will contribute to the establishment and operation of a worldwide active networks and services test bed. The proposed architecture and enterprise model of the initial FAIN specifications makes possible the development, provision and validation of a novel Active Networks architecture for future IP networks. The trials envisaged in the FAIN project will demonstrate inter-connectivity across a worldwide active networks infrastructure in a multi-provider multi-domain environment.

8 Acknowledgements

This paper describes work undertaken and in progress in the context of the FAIN – IST 10561, a 3 year project during 2000-2002. The IST programme is partially funded by the Commission of the European Union.
The FAIN consortium consists of University College London- UK, Jozef Stefan Institute- Slovenia, NTUA- Greece, Universitat Politecnica de Catalunya- Spain, Deutsche Telekom Berkom- Germany, France Telecom/CNET- France, KPN- The Netherlands, Hitachi Europe Ltd.- UK, Hitachi, Ltd.- Japan, Siemens AG - Germany, ETH- Switzerland, GMD Forschungszentrum Informationstechnik GmbH- Germany, IKV++ GmbH- Germany, INTERGAsys – Spain, University of Pennsylvania- USA.

References

1. K. Arnold, J. Gosling "The Java Programming Language" Java Series, Sun Microsystems, 1996, ISBN 0-201-63455-4

2. D. Scott Alexander, Jonathan M. Smith –The Architecture of ALIEN in "Active Networks"- Proceedings First International Working Conference, IWAN'99 - Berlin June 1999, Springer ISBN 3-540-66238-3

3. Biswas, J., et al., "The IEEE P1520 Standards Initiative for Programmable Network Interfaces", IEEE Communications, Special Issue on Programmable Networks, Vol. 36, No 10, October 1998. http://www.ieee-pin.org/

4. J.Biswas, J-F Hurd, A. Lazar, K. Lim, A. Smahjoub, L-F.Pau, M.Suzuki, S.Torstensson, W.Weiguo, S.Weinstein –White Paper on Application Programming Interfaces for Networks, Working Group for IEEEP1520, January 1999.

5. L.K. Calvert - "Architectural Framework for Active Networks, Version 1", Active Network Working Group, Draft, July 1999.

6. CORBA www.corba.org

7. DARPA Active Networks Programme – www.ito.darpa.mil/researck/anets

8. D. Decasper, G Parulkar, S Choi, J DeHart, H Wolf, Bernhard Plattner "A Scalable, High Performance Active Network Node"; IEEE Network, Vol. 13, No. 5, pages 8-19, January 1999.

9. S. Denazis, K. Miki, J. Vicente, A. Campbell – "Designing Interfaces for Open Programmable Routers" in the Proceedings IWAN'99 - Berlin June 1999, Springer ISBN 3-540-66238-3

10. FAIN project WWW Server – May 2000 https://face.ee.ucl.ac.uk/fain/

11. A. Galis, D. Griffin, W. Eaves, G. Pavlou, S. Covaci, R. Broos – "Mobile Intelligent Agents in Active Virtual Pipes" – in "Intelligence in Services and Networks" – Springer Verlag Berlin Heildelberg, April 2000, ISBN 1-58603 007-8

12. A. Galis –"Multi-Domain Communication Management Systems"- CRC Press, July 2000, ISBN 0-8493-0587-X

13. IETF "An architecture for Differentiated Services" S. Blake, August 1998 http://search.ietf.org/internet-drafts/draft-ietf-diffserv-arch-01.txt

14. IKV++ Grasshopper - www.ikv.de/products/grasshopper

15. A. C. Gunter, Scott M. Nettles, J.M. Smith –"The SwitchWare Active Network Architecture –November 1997, IEEE Network Special Issue on Active and Controllable Networks, vol. 12 no.3 pp22-36

16. A. Juhola, I. Marshall, S. Covaci, T. Velte, M. Donohoe, S. Parkkila – "The Impact of Active Networks on Established Network Operators" in the Proceedings First International Working Conference, IWAN'99 - Berlin June 1999, Springer ISBN 3-540-66238-3

17. Van den Merwe, J. E., et al., "The Tempest – A Practical Framework for Network Programmability" IEEE Network, Vol. 12, No 3, May/June 1998

18. MultiserviceSwitching Forum (MSF). http://www.msforum.org/

19. L. Peterson- "NodeOS Interface Specification", AN Node OS Working Group, January 2000

20. J. Postel – "Internet Protocol" – Internet RFC 791, 1981

21. PLAN and PLANETS papers – www./cis.upen/~switchware/PLAN

22 Report of the Sixth Strategic Review Committee on European Information Infrastructure- www.etsi.org/specrec/SRC6/SRC6.htm

23. SwitchWare papers & homepage www.cis.upenn.edu/~switchware

24. TINA www.tinac.com

25. D. Tennenhouse, D. Wetherall – "Towards an active network architecture" Computer Communications Review, 26, 2 (1996), pp 5-18

26. D. J. Wetherall, J. Gutter, D. Tennenhouse – "ANTS A toolkit for building and dynamically deploying network protocol" IEEE OPENARCH, April 1998.

27. World Wide Web Consortium, Extensible Markup Language www.w3.org/XML