Middleware for the management of a large heterogeneous programmable network: a progress report

Andrew Hughes
Department of Computing Science
University College London
Gower Street, London, WC1E 6BT
a.hughes@cs.ucl.ac.uk

Abstract

The interaction between BTExact Technologies and the Department of Computing Science at University College London is becoming increasingly beneficial for both parties. Over the last academic year, there has been a good deal of development in the area of middleware for the management programmable networks. This paper describes the work that has been done, and outlines the plans for future research.

1 Executive Summary

The PhD project I have undertaken is being carried out in conjunction with both University College London (UCL) [44] and BTExact Technologies [43]. My research goal is to create a system that enables a corporate network to reflect characteristics described by a non-static business model. This will be achieved using highly customisable routers based on research done in the Promile [3] project at the department of computing science at UCL, and by providing a management middleware infrastructure that controls the configuration of heterogeneous routers according to declarative policies.

There are two complementary parts to the Promile project. The Promile routing engine is a highly customisable system designed to process IP packets. Promile modules installed in the router are linked together to for a module graph which describes the packet processing procedure. The Promile configuration system is designed to be used by an administrator to program the Promile routing engine. Using the configuration system, router modules can be installed, removed and modified; also the module connection can be manipulated to modify the router engines functionality. The Promile configuration system allows administrators to submit management policies either using XML or a custom graphical language that is translated into XML by the configuration system.

The remaining eighteen months of this project will be spent looking into three areas: a declarative policy language specifications, a policy compiler, and the integration of the compiler with Promile technology using a delivery middleware. The declarative language will be used by an administrator to configure the programmable network according to a corporation's business model; the administrator does not express desired configurations of individual routers, rather the behaviour of the network as a whole is described. The compiler, given a configuration policy defined in the declarative language, generates platform specific messages that are used to configure individual programmable routers. To integrate the programmable network management middleware with Promile technology, an existing delivery system will be used – and possible extended – to enable the compiler to send platform specific policy messages to individual programmable routers. It is expected that the majority of the remaining work in this project will bee in the e design of the compile r.

The final six months of my work will be used to document my findings in the form of a thesis. By the end of my PhD project – in addition a thesis – I intend to have implemented a demonstratable system that can be used to show that my hypothesis is correct; i.e. a system will be implemented that enables a corporate network to reflect characteristics described by a non-static business model.

My work will be evaluated by building the proposed system and deploying it over a testbed. This testbed will be comprised of a number or nodes running the management middleware that extends the work done in the Promile project. Using the management middleware and traffic generation and monitoring tools, it will be shown that the network can be controlled with declarative policies derived by administrators from business models.

2 Background and motivation

The Internet started out as a research project, funded by the Defence Advanced Research Projects Agency [1], called ARPANET [2]. The project was focused on best-effort routing mechanisms that were designed and implemented in the hope that it would still be possible for military computers to communicate in the event that – due to war – some communication routes became unavailable. Routers were equipped with mechanisms to ensure data arrives at the intended destination via any possible network route, thus circumventing network problems. The Internet's use has since been through a number of evolutionary cycles; it now supports a multi-billion dollar industry mainly comprising media delivery and e-commerce businesses. Although the Internet's use has drastically changed since its early days, the fundamental data routing technology has changed very little. For the vast majority of Internet communications, data is transmitted in a best-effort manner. Communications generally have no relationship to business models: data that generates large revenues does not have a greater precedence that data that yields little or no revenue. E-businesses are keen to control data transport in corporate networks such that they reflects their business models; clearly they require a technology that allows them to do so.

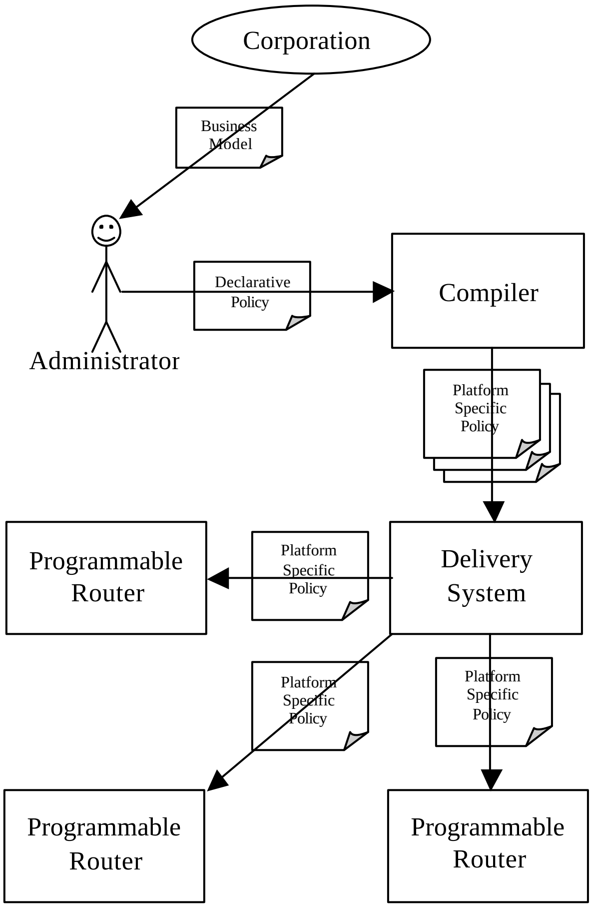
The few corporations that control congestion using router functionality usually make use of Differentiated Service (Diffserv) [37] or Integrated Service (Intserv) [38]. In these paradigms, packets are classified into service levels derived from a business model: gold, silver and bronze, for example. Intserv traffic controllers allocate static bandwidth quotas to each service level. The more advanced Diffserv approach taken by projects such as TEQUILA [41], do not use bandwidth quotas per se; rather than making guarantees about performance, service levels are guaranteed. Differing levels of performance are allocated to each service level and the Diffserv mechanism ensures that each service level performs better than lower service levels.

There is an increasing demand for routers to perform highly customized operations defined by its owner, but commercial off the shelf (COTS) routers are not extensible enough to satisfy this need. COTS routers can be configured to control data flow but they can only use functionality built into router by the vendor. COTS routers are typically configurable but not programmable. By using programmable network routers, the process of routing data can be controlled by the network owner in a highly flexible and customisable manner; this allows Internet corporations to link data transport policies to their business model.

From my research thus far, it has become apparent that a management middleware is required to allow the efficient and correct configuration of a programmable network comprising multiple programmable routers. In this project a middleware is proposed that abstracts individual router configuration from the administrator: the administrator instead controls the system using network scoped configuration policies and the middleware takes the responsibility of configuring individual routers.

Since the middleware can configure multiple nodes simultaneously, there are two heterogeneity issues. The first relates to the architectures of the target routers, the second relates to the target routers' configuration; these are respectively termed syntactic and semantic heterogeneity. To solve these heterogeneity issues, the system will be designed such that the administrator need only produce a declarative configuration policy to describe the desired network behaviour. This policy does not describe the configuration process, rather the state of the system after the management operations have completed. It is the responsibility of the middleware to translate declarative configuration policies into platform specific configuration policies that describe the internal configuration of the programmable routers. From the administrators perspective, the use of declarative policies makes the programmable network heterogeneous in terms of both policy syntax and semantics.

The above system results in the network behaviour reflecting a business model as follows. Given a business model, an administrator creates a network configuration in the declarative policy language and submits it to the compiler. Using network profiling techniques to discover the networks current behaviour, the compiler generates platform specific configuration messages for individual routers that require configuration for the network to reflect the business model. Using a delivery system based on third party message oriented middleware, the compiler delivers the platform specific configuration messages to the programmable routers. The Promile [3] configuration system on these programmable routers interprets the configuration policies and makes the appropriate changes to the Promile packet routing engine. When all the programmable routers have been configured, the network behaviour will reflect the business model. The diagram below illustrates the management system's architecture.



As previously stated, the declarative network configuration policy is derived from a business mode by an administrator, however the term 'business model' has thus far not been defined. In the context of this project, a business model is taken to mean a description of the products and services a company offers; in addition it may includes the company's product pricing structure. For example, a company that offers online movies may have a business model that describes a two levels pricing strategy: one dollar for a low quality video stream and two dollars for a high quality stream. By using a streaming format such as Real Media [39] that adjusts the image quality and frame rate according to available transport speed, the two levels of service can be achieved simple by controlling the bandwidth settings on one of the corporation's routers. Deriving an declarative network configuration policy from an informally stated business model is a difficult task for a computer (and beyond the scope of this project), but for a human administrator the translation is likely be fairly trivial. In our example, an administrator could define four service levels: High quality stream video is platinum, online payment is gold, low quality video is silver and all catalogue browsing (i.e. website access) is bronze. Given these service levels, the administrator may decide to use Diffserv to ensure that each traffic has precedence over less important traffic.

Since the routers in the programmable network are configured according to existing network behaviour, the management middleware should include some reactive management mechanisms. If some routers or physical links within the programmable network fail, in order for the network to continue to function with the desired behaviour (derived from the business model), the remaining functioning routers will have to be reconfigured to tolerate the changes in network structure. In addition, if new routers or physical links are added to the network, the network must be reconfigured; for the sake of simplicity it has be assumed that the management middleware will become aware of additional hardware through administrator interaction.

3 Deficiencies in existing work

Modern day network monitoring and management relies heavily on Simple Network Management Protocol (SNMP) [40]. This protocol was designed to allow remote administrators (either human and computer) to manage heterogeneous network nodes using a lightweight set of command line instructions. This protocol has been integrated into many popular software packages and adopted as the standard remote management protocol by most COTS router vendors. Despite the fact that the use of SNMP in the management of routers has stood the test of time, its usefulness in the management of programmable networks is questionable. SNMP is a protocol that would be extraordinarily difficult to extend to include transactional functionality, security and module installation; and due to the extensibility of programmable network devices, the SNMP low level configuration language is not suited to the management of a large complex network.

Although there are a number of projects that use declarative management mechanisms, to my knowledge there is no other research in industry or in academia that is concerned with the management of heterogeneous programmable networks using declarative configuration policies. PONDER [42] is a declarative language which attempt to standardise the way distributed object enterprise concepts are specified. It does not address some of the areas needed to manage programmable networks, so the work is not suitable to use as the programmable network management middleware.

Nestor [4] and DEN [5] are concerned with the configuration of network; only the former is concerned with programmable networks. Neither of these projects abstract the network configuration away from the configuration of individual routers; to my knowledge, with the exception of my work there is no other project attempting to do this, especially in the area of programmable networks.

The area of programmable networks is relatively new, even so, there are numerous research groups focusing their attention in the domain. Originally, there were two distinct approaches to programmable networks: OpenSig [7] and Active networks [8]. Most of the original OpenSig projects seemed to be striving towards creating a standardized interface to router and switch fabrics; however, industry seems loathed to adopt open interface standards meaning that much of the original work in the OpenSig community has been abandoned. Nowadays, OpenSig projects seems to mainly focus on the control of telecommunication devices such as ATM switches but the level of extensibility (and hence programmability) or OpenSig fabrics are generally not considered to be particularly high; this resulted in the emergence of the active networking approach to programmable networks. Active network projects are typically focused in the area of IP data routing; 'active code' is executed in execution environments that control how packets are processed. Most projects seem take the 'active packet' approach in order to achieve programmability. In this paradigm – in addition to the usual content – data packets include short programs that control the way in which active nodes process packets. Active packets control their own destiny and the active routers merely follow the instruction embedded in the received active packets.

It is becoming increasingly difficult to differentiate between OpenSig and active network projects, this is because projects seem to 'mix and match' traits common to each domain in order to construct a new breed of router. Recent trends indicate that there is a focus shift towards application level programmable networks. In this paradigm, the programmable routers' functionality is controlled by highly customised routing modules (e.g. schedulers, droppers and markers) that are implemented and installed on the router in conjunction with other modules within user-space, all of which are connected into a router module graph that controls the packet processing process.

The work being done in the Promile [3] project falls into the field of application level programmable networks. The project is divided into two parts: a routing engine that processes packets; and a configuration system (based on Xmile [36]) used to configure the router. Both Promile components are have mature designs; test have shown that Promile implementations are efficient when compared to existing routing technology. The Promile configuration system makes changes to the Promile routing engine according to XML [9] policies supplied by an administrator. These policies are syntactically heterogeneous, however they do not abstract router semantics from the administrator. Configuration policies are host specific: they comply with a schema that defines a routers capabilities. The configuration language is not abstract enough to be used to configuration a network, a new language is required to achieve the level of scalability desired for this project. Click [19] is the project most similar to Promile; the main difference between them is that Promile is geared more towards on-the-fly router management. Also in the area of application level programmable networks are the Alpine [6] and Android [18] projects; these projects are closely related but the research seems focused on router architecture and functionality rather than in the area of router configuration.

As previously implied, it is my intention to create management middleware capable of configuring an entire programmable network given a single declarative configuration policy. The Promile project does not take this approach. The XML configuration policies are host-centric; each message is concerned only with the configuration of a single host. This is a major flaw since the 'best effort' routing algorithms built into most routers are likely to interpret some congestion mechanisms as broken data routes, resulting in configurations being autonomously circumvented by other routers. In order for administrators to have complete control the network, they must not have control of the internal configurations of individual programmable routers.

Since it is my intention to create a system to manage an entire programmable network with a single declarative configuration policy derived from a business model. It is sensible to integrate the management middleware with the Promile architecture using a delivery system that can send to multiple destinations in a logically simultaneous manner. Although this could be achieved using asynchronous object oriented middleware, I have reservations regarding runtime extensibility – since both the configuration and the installed software modules will change at runtime – and have therefore decided that use message oriented middleware (MOM) should be used to address the issue. MOM has evolved from a research area into a multi-million dollar industry; clearly this technology is mature and thus suitable for use in this project.

From the discussion so far, it should be clear that one of the key components of the management middleware is the compiler that translates a declarative policy into platform specific configuration policies. In order for the compiler to build platform specific configurations from a declarative policy, knowledge of the network topology and configurations are essential. Since the compiler's dependencies are likely to be highly dynamic, the translation process is likely to be significantly more complex than that of static document translators such as XSLT [10], so although existing document translators may be used to build the management middleware's compiler, they will solve few issues in the compiler problem space.

4 Hypothesis

4.1 Hypothesis statement

I hypothesize that it is possible to program heterogeneous networks using a declarative language according to a business model. We take the view that to achieve the desired level of network extensibility, programmable routers should be used to control data transport over the network. I do not think that there is a need to design and implement a new programmable network technology, rather, I will reuse work that has already been done in that domain. The Promile routing engine will be utilised, in addition the Promile configuration system. I intend to extend the work done in the Promile project by combining it with a policy compiler and a delivery system. The compiler, given a declarative configuration policy, will determine which routers are to be configured and generate platform specific policies to perform the appropriate management operations on each router (it is worth noting that there is a fairly high likelihood that the configuration of each programmable router will be different). The delivery system will deliver platform specific message to Promile configuration system residing on each of the programmable routers needing altering. This configuration system will then modify the Promile routing engine as appropriate. Since the middleware allows the configuration of multiple nodes, it is likely that management operations will need to be done with transactional properties; the middleware will therefore include this functionality.

In order for a programmable network to be managed using only network scoped declarative configuration policies, there must be a language that is expressive enough to reflect most desired network configurations yet have a level of abstraction such that configuration messages are semantically heterogeneous with regard to the configuration of programmable routers; this language will be defined in this project. The level of expressiveness must be such that the administrator has complete control of the network behaviour, allowing custom router modules to be used to affect the network performance. The level of abstraction should ensure that the administrator cannot configure individual routers, therefore avoiding the possibility that some desired network behavioural rules are not enforced.

The compiler, which generates platform specific messages from network scoped configuration policies written in the declarative language defined in this project, is likely to rely heavily on profiling tools that discover the existing network behaviour. From this profile, the compiler calculates the required platform specific changes outlined in the declarative configuration policy. The compiler is therefore a very significant part of the management middleware.

4.2 Hypothesis testing 1

4.2.1 Proposal

A declarative configuration language will be defined that is sufficiently expressive to allow an administrator to configure a programmable network yet abstract enough to allow the middleware to control individual routers. For the hypothesis to be confirmed as correct. It must be verified that the declarative language can be used to configure the network to reflect all reasonable configurations. The definition of such a declarative language is especially difficult since – to control the network traffic – it must be capable of defining the router modules that are installed in the programmable routers without describing the configuration of routers. The language must be appropriate to express network configurations associated with business models.

4.2.2 Rational

Since it is intended that network administrators need only use the declarative language defined in this project configure the network, it is clear that configuration policies written using the language must be accurately represent the desires of the administrator. Furthermore, it must be possible for this language to be processed by the compiler to generate platform specific messages that are used to configure individual programmable routers. The language must therefore exhibit appropriate levels of abstraction and expressiveness so as to be used as intended. If the language is not expressive enough, it will not be possible for administrators to configure the network such that it reflects a business model. If the language is not abstract enough the administrator is likely to produce network configuration policies that cannot be enforced by the programmable router.

4.2.3 Assumptions

To verify that the language can be used by an administrator to configure a programmable network, it must be shown that programmable networks can be configured to reflect business models using the declarative language. To achieve this, a sufficient number of business models relating to network configurations must be acquired, it must then be shown that the language can be used to represent the model. It is assumed that enough quality network configurations can be derived from example business models; and that the business models are representative of real corporate network requirements. The business models will be acquired through my interactions with BTExact; the desired network configurations will be derived from these business models by myself, my research group and by BTExact.

4.2.4 Methodology

The language must be analysed to ensure that it can represent all configuration policies that are realistically likely to be desired of the network without describing host specific configuration details. I do not believe that the correct approach to the analysis uses automated tools; rather, a scientist can verify the language by inspection. The language verification process will require the acquisition of an extensive range of network configurations that are likely to be requested. It is my intention to define the language such that all the example network configurations can be represented. The testing procedure is not a process separate work package to the language definition; the declarative language design will involve the use of numerous example network configurations.

To show that the language can be used to manage a live corporate network, the management system – comprising the declarative configuration language – will be deployed on an example corporate network that accurately reflects a live business system. Using traffic generators and network monitoring tools, it will be shown that programmable network configurations expressed in the declarative language can control the network's behaviour as expected. By demonstrating the functionality of the management system in this way, it can be verified that the language is suitable for the use in the programmable network management middleware.

4.2.5 Predictions

Given that the declarative language will be developed using example business models, I am confident that all the example network configurations can be represented using the declarative language defined in this project. As previously stated, it has been assumed that example business models can be acquired that are sufficiently representative of real corporate network configuration requirements; since it is expected that the language is suitable for use in all the example scenarios, the language will be capable of representing most (if not all) network configurations derived from real business models. This will mean that the declarative language is sufficiently expressive, yet adequately abstract, to configure a programmable network according to a business model.

4.3 Hypothesis testing 2

4.3.1 Proposal

Given a declarative configuration policy derived from a business model by an administrator, the compiler can generate configuration messages for each heterogeneous programmable router, resulting in a network configuration that reflects a business model. We must verify that the policies generated by the compiler to configure a network adequately reflect the desired network behaviour described by the declarative configuration policy. The platform specific messages generated by the policy compiler must then be shown to be delivered to the appropriate programmable networks, using the management system's embedded delivery system, with sufficient efficiency and with relevant functional behavioural properties (such as transactions and security).

4.3.2 Rational

The key goal of this project is to create a system that can configure a programmable network according to an declarative policy supplied by an administrator. For the system to be accepted by industry, this goal must be met. Clearly, if the configuration of the network does not reflect the policy given by the administrator, the compiler is not sufficiently functional to correctly configure the network, therefore there is little point in using the management middleware. The platform specific messages generated by the declarative policy compiler are used to configure the programmable network routers; these platform specific policies, taken together, must therefore represent the requirements outlined in the declarative policy.

For the platform specific messages to correctly configure the programmable network, there is a clear need for the delivery systems to be sufficiently functional and efficient to be used to configure the network. The delivery system is the 'glue' that enables the programmable routers to draw on the functionality provided by the language compiler.

4.3.3 Assumptions

The language compiler will be implemented so that the functionality can be verified. The compiler relies heavily on mechanisms that are capable of determining a network's behaviour. It is not the intention of this project to focus heavily in this area, rather, existing research will be drawn upon and existing tools will be reused in the programmable network management system. It is assumed that appropriate research and tools are in existence and are available for use.

It has previously been emphasised that the a robust delivery system is needed to deliver platform specific messages to programmable routers. Research into delivery middleware mechanisms is beyond the scope of this project; it is assumed that there exists an available delivery system that is appropriate for reuse in the this project.

4.3.4 Methodology

To test if the compiler is correctly deriving a platform specific configuration messages from a declarative network policy, a network profile will be built. The profile will consist of multiple statements – written in the management middleware's declarative configuration language – that describe the network's behaviour. If the declarative policy being tested matches one of the policies in the network profile, then the configuration generated by the compiler will be assumed to be correct.

The management middleware – together with the Promile technology that it extends – will be deployed over a testbed in order to test that the network is managed correctly. If the network does not exhibit the expected behaviour following the configuration process, the compiler is likely to be flawed.

The testing of the functionality and efficiency of the delivery middleware will be done during the selection process of the message oriented middleware that is to be integrated into the programmable network management middleware. I believe this the testing need not comply with a scientific methodology. From sensibly selecting and experimenting with potential delivery systems, it will be a fairly trivial task to determine which to use to integrate the management middleware with the routing fabric. To prove that the integration is a success the system should be demonstrated to work on a live network. In this way, the system can be shown that the network behaviour can reflect a business model.

4.3.5 Predictions

I am confident that a compiler can be constructed that is capable of generating the correct set of platform specific messages. Given that both the compilation process and the testing process depend on the same network profiling mechanisms, for the system to be correct accurate network profiles must be created. I expect that, with sufficient research, this can be achieved.

The programmable network management middleware is to be integrated with the programmable network routers using a delivery middleware, I expect that a delivery system can be found that is suitable for this.
It is my belief that the testing procedures are sufficient to verify the correctness of both the compiler and the delivery system.

4.4 Hypothesis testing 3

4.4.1 Proposal

In order for the management middleware to be classified as a useful system and adopted by industry to manage a corporate network employing programmable router technology, the performance of the system must be shown to be far better than other methods of programmable network management. Since this is the only project that is concerned with creating a middleware for the management of a programmable network using declarative configuration policies derived from business models, the system can only be compared with programmable network management systems that do not use declarative configurations. The few programmable network projects that are capable of configuring an entire network comprising programmable network routers will be analysed and shown to be less functional than the management system proposed in this project.

4.4.2 Rational

For a corporate network to benefit form the extensibility of programmable networking technology, it must be possible for the routers to be controlled in a reliable and efficient manner. This project aims to provide a middleware that allows this. If it can not be shown that there are significant benefits from using the management system proposed in this project then it is unlikely to be deployed on a live corporate network. This project is funded by both academia and industry, and therefore needs to incorporate sound scientific ideas with realistic business opportunities. By comparing and contrasting the functionality of this system with similar existing technology, the value of this work can be demonstrated to be accepted in both of the domains.

4.4.3 Assumptions

In order to contrast the management middleware proposed in this project with other programmable network configuration middleware, numerous example configuration scenarios must be identified that can be achieved using all of the management systems. This project assumes that these example business models can be acquired through my interactions with the corporate sponsor. BTExact continues to produce significant research that is of interest to both industrial and academic bodies, and it therefore has access to business models that can be utilised by this project.

4.4.4 Methodology

Before this testing process can be undertaken, a survey of existing projects must be carried out to identify the relevant systems that can be compared to the programmable network management middleware proposed in this project.

Using both the management middleware proposed in this project and similar configuration systems, identical networks will be created using each technology. The example business models acquired form the industrial sponsor will be given to a network administrator who will modify the network accordingly. The configuration process required using each configuration system will be identified; the ease of use and network behaviour correctness of each configuration system will be contrasted. At present, it is thought that the most likely systems to be included in the comparison are Nestor [4] and DEN [5]. It is believed that these are the closest related projects to my work.

4.4.5 Predictions

It is expected that, compared to similar projects, the management middleware proposed in this project will improve the way in which programmable routers are managed. The declarative configuration policies describe the behaviour of the network rather than individual routers and the compiler determines the configuration of each router; this means that human configuration error at the router level is eliminated. The declarative approach to configuration specification prevents humans from describing the configuration process, human error is also eliminated here. The system proposed in this project includes intelligence that, until now, has been provided by the network administrator. To my knowledge this is the only project that takes this approach to the management of programmable networks, so is likely to be better suited to programmable network management than similar systems.

4.5 Conclusion

The remaining research needed to produce a programmable network management middleware by extending work done in the Promile project falls within three areas:

- The definition of a declarative policy language
- The design of a policy compiler
- The integration of the programmable network management middleware with Promile technology using a delivery middleware

As can be seen from the description of tests that will be performed to verify my hypothesis, each and every one of the above areas will be addressed. The declarative language will be verified; the functionality of the policy compiler will be show as correct; and the integration of the management system with the Promile programmable router technology will be shown to be of an acceptable standard. In addition to these tests, the system proposed in this project will be compared to similar network configuration systems. It will be shown that the management system can manage a network to better reflect a business model than the other systems.

It is my aim to produce a system that can be deployed by the project’s corporate sponsors onto a live corporate network. In terms of academic acceptability, I am confident that the quality of the work and testing process will add to the scientific communities knowledge and be sufficient to warrant continuing research on the project. The tests outlined in the previous sections seem to be adequately complete and correct for this to occur.

5 Related work

The work being done in the area of programmable routers is extensive. Although the OpenSig [7] community no longer seems likely to define a standardized open interface to routers and switches, the PIN committee [11] is still in existence and projects such as X-Bone [12] and Mobiware [13] still generate interest. In the area of active networking [8], research in projects such as ANTS [14] and Active Services [15] are still flourishing. For a survey on OpenSig and active network projects refer to [16] and [17]; these papers contain extensive descriptions and similar works are contrasted. As stated previously, my research extends work carried out in the domain of application level programmable networks. The key projects in this area are Promile [3], Android [18] and Click [19].

There are few projects that are concerned with the management of programmable networks: NESTOR [4], SENCOMM [20], ABLE [21] and ANCORS [22] are among them. These projects seem to management systems for programmable routers rather than complete networks. One of the key components of the programmable network management middleware is the compiler that translates a declarative configuration policy into host specific configuration policies based on network particulars and existing router configuration; this is relevant to work being done by the Distributed Management Task Force [23], more specifically the Directory Enabled Network (DEN) initiative [5]. In fact, the Intelliden corporation [24] is bringing this technology to market place.

The compilation process is partly relevant to the work being carried out in the Semantic web [25] domain. Using the Resource Description Framework, the W3C [26] are working towards creating mechanisms that allow data to be interpreted to suit individual interests. With respect to the network monitoring capabilities required by the compiler, work done on network performance measurement tools similar to PathChar [27] and the Network Weather Service [45] in addition to work completed on router monitoring tools such as Ganglia [28] and MRTG [29] can be drawn upon.

In terms of delivery middleware, there has been much work done; in fact, there are numerous technologies that are commercially mature. The main design goal of the Java Messaging Service [30] (part of the Enterprise Java [31] specification) seems to be allowing web applications and enterprise beans to communicate; however, its use as a general purpose message oriented middleware is also well known In the financial sector, Tibco's Rendezvous [32] seems to be the most popular message oriented middleware, however there are two significant competitors, MQSeries [33] and MMQ [34], made by IBM and Microsoft respectively. All these systems include transaction and security services.

To my knowledge, this project is unique. There does not seem to be any other individual or research group attempting to create a programmable network management middle using network scoped declarative configuration policies.

6 Work to date

It is expected that students in the Department of Computing Science at UCL should spend three years working towards their PhD, I am confidant that I will complete my research goals within this time frame. I am now nearing the end of my first year. The last twelve months have been used to determine an interesting problem that is suitable for a PhD thesis and to read sufficient background material as to justify the relevance of my work to current research trends. Hitherto, my reading has mainly been in the following areas: programmable networking (application level, OpenSig and Active networking); middleware (mainly message oriented middleware, but also object oriented and agent middleware); network protocols (in particular reliable multicast and application layer peer-to-peer messaging); application servers (J2EE); and XML (including XPath, XSLT, Schemas and DTD). Previously the focus of my work was in the area of delivery middleware. I wrote and published a paper [35] describing the ideal requirements for such a system. More recently my research has moved towards network management. I feel that I have now done sufficient research to focus on the definition of the declarative language and the design of a compiler.

7 Timescales

The next eighteen months will be broken down as follows. The next three months will be used to define the declarative language used by administrators to manage the network. The following six months will be spent designing the compiler, including the design network profiling mechanisms. Three months will be taken to implement the compiler once the design is complete. I believe that once the compiler has been implemented, three months will be sufficient to integrate the management middleware with Promile technology. The remaining six months of my project will be used to write the PhD thesis. This time includes two months for system testing and four months to concentrate on the documentation of my work.

8 Bibliography

[1] DARPA, http://www.darpa.mil/.

[2] Clark, D. D. "The Design Philosophy of the DARPA Internet Protocols", in the proceedings of ACM SIGCOMM '88, August, 1988.

[3] M. Rio, N. Pezzi, H. De Meer, W. Emmerich, L. Zanolin, C. Mascolo. "Promile: A Management Architecture for Programmable Modular Routers", in the proceedings of OpenSIG 2001, September 2001.

[4] A. V. Konstantinou, Y. Yemini, and D. Florissi. "Towards Self-Managing Systems", in the Proceedings of DARPA Active Networks Conference and Exposition (DANCE), May 2002, San Francisco, CA.

[5] Directory Enabled Networks initiative. http://www.dmtf.org/standards/standard_den.php

[6] I. Marshall, J. Cowan, J. Crowcroft, M. Fry, A. Ghosh, D. Hutchinson, D. Parrish, I. Phillips, M. Sloman, D. Waddington, "Alpine - Application Level Programmable Inter-Network Environment", BT Technology Journal 15(2), April 1997.
[7] OpenSig, http://www.ctr.columbia.edu/opensig/.
[8] DARPA Active Network Program, http://www.darpa.mil/ito/research/anets/.
[9] XML, http://www.w3.org/XML/.
[10] XSLT, http://www.w3.org/Style/XSL/.
[11] J. Biswas, et al. "The IEEE P1520 Standards Initiative for Programmable Network Interfaces", IEEE Communications Magazine, Special Issue on Programmable networks, October 1998.
[12] J. Touch, S. Hotz. "The X-Bone". Third Global Internet Mini-Conference in conjunction with Globecom '98, Australia, November 1998.
[13] O. Angin, A. T. Campbell, M. E. Kounavis, R. R.-F. Liao, "The Mobiware Toolkit: Programmable Support for Adaptive Mobile Networking", IEEE Personal Communications Magazine, August 1998.
[14] D. Wetherall, J. Guttag, and D. Tennenhouse. "ANTS: A Toolkit for Building and Dynamically Deploying Network Protocols". In the Proceedings of IEEE OPENARCH '98, April 1998.
[15] E. Amir, S. McCanne, and R. Katz, "An active service framework and its application to real-time multimedia transcoding", in proceedings of ACM SIGCOMM '98, September 1998.
[16] A. T. Campbell et al., "A Survey of Programmable Networks", in the proceedings of ACM SIGCOMM Computer Communication Review, April 1999.
[17] I. Marshall, et al., "Application-level programmable internetwork environment", BT Technology Journal 17(2), April 1999.
[18] I. Liabotis, O. Prnjat, L. Sacks, "Policy-Based Resource Management for Application Level Active Networks", in the proceedings of the 2nd IEEE Latin American Network Operations and Management Symposium (LANOMS), Brazil, August 2001.
[19] E. Kohler, R. Morris, B. Chen, J. Jannotti, M. Kaashoek, "The Click modular router", in the proceedings of ACM Transactions on Computer Systems 18(3), August 2000.
[20] A. Jackson, et al., "Active Network Monitoring and Control: The SENCOMM Architecture and Implementation", in the proceedings of DARPA Active Networks Conference and Exposition (DANCE), USA, 2002.
[21] D. Raz, Y. Shavitt, "An Active network Approach for Efficient Network Management", in the proceedings of IWAN, Germany, 1999.
[22] L. Ricciulli, et al. "An Adaptable Network Control and Reporting System (ANCORS)", in the proceedings of DARPA Active Networks Conference and Exposition (DANCE), USA, 2002.
[23] DMTF, http://www.dmtf.org/.
[24] Intelliden, http://www.intelliden.com/.
[25] Semantic web, http://www.w3.org/2001/sw/.
[26] World Wide Web Consortium, http://www.w3.org/.
[27] A. Downey, "Using PathChar to estimate Internet link characteristics, in the proceedings SIGCOMM 1999, Cambridge, MA, September 1999.
[28] Ganglia, http://ganglia.sourceforge.net/.
[29] Multi Router Traffic Grapher, http://people.ee.ethz.ch/~oetiker/webtools/mrtg/.
[30] JMS, http://java.sun.com/products/jms/.
[31] J2EE, http://java.sun.com/j2ee/.
[32] Tibco, http://www.tibco.com
[33] MQSeries, http://www-3.ibm.com/software/ts/mqseries/.
[34] MMQ, http://www.microsoft.com/msmq/.
[35] A. Hughes, "Requirements of a Middleware for Managing a large heterogeneous programmable Network", BT Technology Journal 20(2), April 2002.
[36] C. Mascolo, L. Zanolin, W. Emmerich "XMILE: an XML based Approach for Incremental Code Mobility and Update", Automated Software Engineering Journal (Special Issue on Mobility). Kluwer.
[37] S. Blake et al., "An architecture for differentiated services", RFC 2475, 1998.
[38] R. Braden, D. Clark, S. Shenker, "Services in the Internet Architecture: an Overview", RFC 1633, 1994.
[39] Real Media, http://www.real.com/.
[40] J. Case, M. Fedor, M. Schoffstall, J. Davin, "A Simple Network Management Protocol (SNMP) ", RFC 1157, 1990.
[41] P. Trimintzios, et al., "A Management and Control Architecture for Providing IP Differentiated Services in MPLS-based Networks", IEEE Communications Magazine, 39(5), May 2001.
[42] E. Lupu, M. Sloman, N. Dulay, N. Damianou, "Ponder: Realising Enterprise Viewpoint Concepts", 4th International Enterprise Distributed Object Computing Conference (EDOC2000), Japan, September 2000.
[43] BTExact Technologies, http://www.btexact.com/ .
[44] University College London, http://www.ucl.ac.uk/.
[45] R. Wolski, "Dynamically Forecasting Network Performance Using the Network Weather Service", Journal of Cluster Computing Volume 1, pp. 119-132, January, 1998.

Andrew Hughes is currently perusing a PhD in the Department of Computing Science at University College London, his research is focused in the area of programmable networks. He is one of the first students to be based at the new UCL campus at Adastral park which is designed to strengthen the collaboration between the university and BTExact Technologies. Andrew graduated from the University of Newcastle-upon-Tyne in 2001 with an honours degree in computing science. In 2000, he co-founded a software development company that works in the area of virtual networking.