DUBLIN CITY UNIVERSITY

School of Electronic Engineering

A Distributed Intelligent Network based on CORBA and SCTP

Rob Brennan B.Sc. M.Sc.

This thesis is submitted for the award of *PhD in Electronic Engineering*.

Name of Supervisor: Prof. Thomas Curran
Submission Date: August 30th 2004

I hereby certify that this material, which I now submit for assessment on the programme of study leading to the award of *PhD in Electronic Engineering* is entirely my own work and has not been taken from the work of others save to the extent that such work has been cited and acknowledged within the text of my work.

Signed:  _____ ID No.: 98970062

Date: August 30th 2004

Table Of Contents

Abstract

Thesis Title: A Distributed Intelligent Network based on CORBA and SCTP

Author: Rob Brennan

The telecommunications services marketplace is undergoing radical change due to the rapid convergence and evolution of telecommunications and computing technologies. Traditionally telecommunications service providers' ability to deliver network services has been through Intelligent Network (IN) platforms. The IN may be characterised as envisioning centralised processing of distributed service requests from a limited number of quasi-proprietary nodes with inflexible connections to the network management system and third party networks. The nodes are inter-linked by the operator's highly reliable but expensive SS.7 network. To leverage this technology as the core of new multi-media services several key technical challenges must be overcome. These include: integration of the IN with new technologies for service delivery, enhanced integration with network management services, enabling third party service providers and reducing operating costs by using more general-purpose computing and networking equipment.

In this thesis we present a general architecture that defines the framework and techniques required to realise an open, flexible, middleware (CORBA)-based distributed intelligent network (DIN). This extensible architecture naturally encapsulates the full range of traditional service network technologies, for example IN (fixed network), GSM-MAP and CAMEL. Fundamental to this architecture are mechanisms for inter-working with the existing IN infrastructure, to enable gradual migration within a domain and inter-working between IN and DIN domains. The DIN architecture compliments current research on third party service provision, service management and integration Internet-based servers.

Given the dependence of such a distributed service platform on the transport network that links computational nodes, this thesis also includes a detailed study of the emergent IP-based telecommunications transport protocol of choice, Stream Control Transmission Protocol (SCTP). In order to comply with the rigorous performance constraints of this domain, prototyping, simulation and analytic modelling of the DIN based on SCTP have been carried out. This includes the first detailed analysis of the operation of SCTP congestion controls under a variety of network conditions leading to a number of suggested improvements in the operation of the protocol. Finally we describe a new analytic framework for dimensioning networks with competing multi-homed SCTP flows in a DIN. This framework can be used for any multi-homed SCTP network e.g. one transporting SIP or HTTP.

Chapter 1. Introduction

In this chapter we provide a brief overview of the problem domain, the objectives of our work, the main contributions of this thesis and finally an outline of the structure of this document.

1.1. Overview

The telecommunications services marketplace is undergoing rapid change. The pace of this change is visible in many aspects of our daily lives; ubiquitous mobile access to digital services, viewer interaction with broadcast media, deregulation of the marketplace, the dot com boom and bust. These are all symptoms of the explosive convergence of telecommunications and computing technologies. Spurred on by the successes of the Internet and the increasing sophistication of consumers of multi-media services, traditional telecommunications service providers must act decisively to secure their place in the new world order. It is easy to visualise a world in which they do not succeed, encumbered by a stolid regularity framework, huge workforces equipped with obsolescent technical skills and enormous capital investment in an obsolescent 19th Century infrastructure. The competitive advantages of new telecommunications operators and manufacturers using information technology and data-communications principals are obvious. Of course this is not the full story. It is the belief of many analysts that apart from obvious advantages, such as brand familiarity and market penetration, the existing providers hold one key *technical* advantage at the heart of their networks – the ability to provide access to reliable, customisable, cost-effective network services through the Intelligent Network (IN) platform.

Unfortunately the current IN model and deployment philosophy is rooted in the technological challenges and methodologies of the 1980s. Thus, IN may be characterised as envisioning the centralised processing at Service Control Points (SCP) of distributed service requests that originate at Service Switching Points (SSP). Supporting service nodes such as Intelligent Peripherals (IP) and Service Data Points (SDPs) add to the distribution of service execution and flexibility of this structure. Each element in this architecture is generally deployed on a centralised server or directly on telephone switching equipment. The nodes are inter-linked by the operator's highly reliable but expensive SS.7 network. To leverage this technology as the core of new multi-media services key technical challenges must be overcome. These include: integration of the IN with new technologies for delivering digital content, more efficient management of network resources, the opening up of the market to allow third

party service providers, a reduction in operating costs by using more general-purpose computing and networking equipment and an expansion of the scope of IN to include service control for hybrid services which encompass both traditional telephony aspects and the use of new technology such as Internet-based resources.

Any future development of the IN model must take all of these driving forces into consideration while retaining interoperability with the deployed service network. This is view is reinforced if you consider that it is likely that any new technology will only be introduced slowly and perhaps initially as isolated islands of enhanced functionality. There is general consensus that previous efforts to replace or radically enhance the IN have universally failed on the issue of interoperability with existing infrastructure.

Consumers, operators and vendors are becoming increasingly familiar with the benefits of interconnection with and re-use of IP-based networking technology in the telecommunications domain. From the consumers' point of view, standard telephony services should be available in and seamlessly integrated with the new IP-domain; from the operators' point of view the exponential growth in demand for IP-based services and cost savings available by using non-specialist equipment are highly attractive. Finally, equipment vendors see that unless they understand and harness these proliferating technologies new vendors will eclipse them in the near future. The current incompatibility between the closed telephony network and the open Internet must be addressed.

In this document we present a general architecture that defines the framework and techniques required to realise an open, flexible, middleware-based distributed intelligent network (DIN). Fundamental to this architecture are mechanisms for inter-working with the existing IN infrastructure, to enable gradual migration within a domain and inter-working between IN and DIN domains. This middleware-based system can be used for the provisioning of distributed service logic in a third party service provider marketplace and also provides an open platform for leverage of resources, such as Internet-based servers, which are isolated from the current Intelligent Network. Given the dependence of such a distributed platform on the transport network that links individual nodes, our approach also includes a detailed study of the emergent IP-based telecommunications transport protocol of choice, Stream Control Transmission Protocol (SCTP). In order to comply with the rigorous performance constraints of this domain, prototyping, simulation and analytic modelling of the DIN based on SCTP have been carried out.

1.2. Objectives

The overall objective of this work is to enable the deployment of next generation telecommunications service platforms, which are capable of both leveraging telecom operators' existing investment in IN and providing a flexible and reliable environment to exploit new service opportunities. This overall objective has been decomposed to identify several key sub-objectives. These are as follows:

- **To survey the literature for next generation services and service platforms**.

  The identification of key trends and requirements for any new service architecture is vital to the ultimate success or failure of that architecture. The next generation service-space is currently in a state of flux with a proliferation of new technical possibilities and business drivers. Thus both traditional telecommunications and newer data communications-based solutions must be considered.

- **To specify a flexible architecture for next-generation service control**.

  Current IN platforms are inflexible and often proprietary in nature. To enable IN evolution, it is necessary to create a flexible service platform specification. This new architecture must be seamlessly integrated with the existing IN and also enable the integration of new service technologies such as IP-based multi-media services. Finally, the architecture should be less dependent on monolithic computing resources to enable a proliferation of small operators and 3rd party service providers.

- **To investigate and evaluate the use of SCTP as a transport protocol for linking DIN nodes in the next-generation service platform**.

  Any new architecture for service control must be able to run on an IP network. SCTP has been designed as transport for signalling over IP and is hence a natural choice for the DIN distributed platform. In such a distributed system the transport protocol used will have a large impact on end-to-end performance. Given this and the sensitivity of signalling traffic to delay and message loss, it is vital to carefully consider the protocol used to transport that traffic.

- **To evaluate the likely performance of the new architecture**.

  Central to the field of telecommunications systems engineering is the performance evaluation of technologies and architectures before they are deployed. It is through this careful engineering of the network that sustainable long-term growth in service capacity will be maintained. Hence it is vital to explore the performance aspects of the new architecture through simulation and analysis.

1.3. Summary of Main Contributions

This thesis addresses the topic of flexible next-generation telecommunications service platforms. Specifically it deals with architecture for and performance modelling of platforms based on CORBA middleware and SCTP signalling transport.

The main contributions are:

- **A new middleware-based architecture for Distributed IN (DIN)**. This architecture enables the evolution of current telecommunications service networks towards flexible distributed service platforms which have stronger integration with new service delivery mechanisms and network management systems. It addresses key weaknesses in the current IN in terms of cost of ownership, flexibility of implementation, isolation from the TMN and dependency on specialised equipment vendors. It enables a wide range of new deployment scenarios and supports current industry trends towards ICT integration and third party service provision. This architecture has been standardised by the OMG Telecommunications Domain Task Force. It incorporates:
  - A survey of the public literature relating to the design and deployment of CORBA-based telecommunications service platforms, together with an indication of some issues not addressed therein;
  - An architectural framework for the implementation of both native CORBA TC-User Application Entities (SSPs, SCPs) and CORBA to TC-User Gateways;
  - A specification translation algorithm for the conversion of ITU ASN.1 TC-User application protocols into OMG IDL object interface specifications;
  - An interaction translation specification that deals with how to name, locate and interact with TC-User implementations in the CORBA domain in a way that provides location transparency for both native CORBA implementations and traditional TC-User implementations in the SS.7 domain;
  - A set of TC PDU-oriented Interfaces that define an API that standardises access to SS.7 protocol stacks in the CORBA domain, either for use at CORBA to TC-User Gateways or by CORBA applications that wish to use TC/SS.7 message transport;
  - A survey of deployment scenarios and implementation experiences;
- **A comprehensive examination and appraisal of the performance of the SCTP signalling transport protocol**. This work has identified a number of flaws with the current protocol's congestion control specification leading to a number of suggested improvements in the operation of the protocol to fix defects and improve performance. These recommendations have been accepted by the IETF and are included in the working draft for the next official version of the specification. In addition we developed the first analytic model for the throughput of a single multi-homed SCTP association. This model has been evaluated over a wide range of network conditions and shown to

produce a reliable estimate of performance on a single association. This work incorporates:

- A survey of the literature relating to aspects of the operation and performance of congestion control as implemented in SCTP and its precursor TCP;
- An analysis of the operation of single-homed SCTP congestion controls using experimental results and both analytic and simulation models under a variety of network conditions;
- The development and testing of an analytic model for the throughput of a single multi-homed SCTP association;

- **Development of a framework for dimensioning of competing multi-homed SCTP flows**. This is the first analytic framework for evaluation of the performance of networks of arbitrary topologies of competing multi-homed SCTP associations. It has been shown to be computationally efficient and capable of rapidly producing results with an acceptable degree of accuracy for network dimensioning tasks. We have also illustrated how it can be combined with a modified version of standard SCP models to produce estimates of DIN application performance. This work incorporates:
  - Enhancements to our analytic model for the throughput of a single multi-homed association to enable the modelling of networks of competing flows;
  - A set of signalling network scenarios for DIN over SCTP networks;
  - An analysis of the performance of networks of DIN nodes utilising multi-homed SCTP associations for signalling transport;

1.4. Structure of the Thesis

The rest of this document is structured as follows:

In chapter 2 we present a background context for our work. First we discuss the historical development of the Intelligent Network and current directions in next generation services research. This allows us to present a coherent vision for the evolution of the IN and identify key technical challenges to be faced. One consequence of this discussion is the identification of middleware as the basis for our work on a service platform. We then outline current research on congestion control and performance analysis of IP transport protocols, specifically SCTP and its predecessor TCP. This will form a necessary background to our detailed treatment of SCTP performance in chapters 5 and 6.

In chapter 3 we give an introduction to the methodologies and technologies that we have employed in the course of our work. This consists of an overview of middleware technologies, a discussion of specification languages such as IDL and UML and background material on analytic and simulation-based modelling techniques.

Chapter 4 provides an in-depth look at our proposed middleware-based architecture. We propose a

CORBA-based solution for service platforms, which addresses our design objectives of increased deployment and service handling flexibility while supporting smooth integration with the existing IN. This new approach is labelled the Distributed IN (DIN). We also show how our approach can be generalised to allow any SS.7 Transaction Capabilities (TC) user protocol (such as MAP, which is used for GSM services) to be supported. We then discuss some possible deployment scenarios and report on our prototyping experiences.

Chapter 5 establishes our framework for studying the signalling transport, SCTP. We first present a new analytic model for modelling multi-homed SCTP association on a single multi-homed network route and then describe our model of SCTP for event-based simulation. Next we investigate the performance and resilience to overload of SCTP in a number of scenarios for a single network route. The work presented here leads to a number of suggested improvements for the behaviour of SCTP when network congestion is encountered. The predictions of our analytic model are also validated by comparison with event-based simulation and experimental results.

In chapter 6 we extend our analytic model for the performance of a single multi-homed SCTP association to model the performance of networks of competing multi-homed SCTP associations. The results of our extended network model are then compared with event-based simulation results for a set of increasingly realistic network scenarios. Finally we illustrate the application of our SCTP network model to DIN network planning tasks by combining our SCTP model with a simple model of our DIN architecture to study system performance.

Finally in chapter 7 we present our conclusions, a summary of our main contributions, new potential applications of the work presented here and suggested topics for further research.

Chapter 2. Background Information and Literature Review

In this chapter we provide a brief overview of the state of the art regarding the evolution of Intelligent Networks (IN) and IP based telecommunications signalling transport. §2.1 gives an overview of the current IN architecture, the driving forces for architectural evolution, emerging technologies and the key issues to be solved in future networks. §2.2 gives an overview of CORBA technology and current research on CORBA-based telecommunications services. §2.3 gives an overview of SCTP, a new protocol for IP-based telecommunications signalling transport. §2.4 provides a comprehensive review of the publicly available literature relating to the performance of IP transport protocols, SCTP and its predecessor TCP. Finally §2.5 draws conclusions from the literature review and outlines the approach described in the remainder of the thesis.

2.1. Distributed Intelligence in Telecommunications Networks

Recent years have seen a huge increase in the prevalence of desktop computing in homes and businesses worldwide. This has been fuelled, to a large extent, by the success of the Internet, which has clearly demonstrated the immense commercial potential of multimedia communications services. Exposure to the Internet has raised customer expectations of the service features that should be offered by the public telecommunications infrastructure, principally that they support a mix of media types and allow easy customisation. Additionally, users expect that these services will be available on demand, regardless of their location or the capabilities of their terminal equipment. Whilst the main technological components required to realise this vision are available, there remains a significant challenge in deploying them in a manner that is both cost effective and that can continue to meet the demands of a volatile marketplace.

As the level of interconnection between fixed, mobile, Internet and enterprise networks increases, a key component in ensuring their ongoing success will be the availability of a common platform for the development and delivery of communications services. Of course a key requirement for operators who intend enhancing their service delivery capabilities is that existing systems are leveraged as much as possible rather than replaced outright. Many see the Intelligent Network (IN), which is today the prevalent means of providing services based on manipulation of voice call set-up, as a starting point for

the service delivery platform of the future. Currently a number of groups are proposing short to medium term evolutionary paths for IN that aim at overcoming limitations of existing systems. In this section we outline the basic elements of the IN as it exists today, discuss some of the technical and commercial limitations which are currently driving the development of IN, describe some of the evolutionary initiatives and identify a number of common trends in the development of IN and outstanding issues for which immediate solutions are not apparent.

2.1.1. The Intelligent Network

The IN provides the PSTN (Public Switched Telephony Network) with the infrastructure to provide advanced services such as freephone (1-800) and number portability. The Intelligent Network came into being in the mid-1980s as a way of de-coupling telecommunications service logic from the call switching functions of exchanges. This facilitated centralised service processing functionality, eased the deployment of new services and reduced the escalating complexity of exchanges. IN standardisation has taken place in ANSI, the ITU and ETSI. Unfortunately this distribution of standardisation effort and the proprietary enhancements to IN by vendors have created a plethora of non-interworking IN solutions. Within geographical regions, for example the USA or Europe, there is sufficient agreement on standards that interworking is possible and, at least nominally, the ITU standard forms the basis for international interworking.

The basic IN model involves a distributed functional architecture which contains functional entities which are collectively responsible for handling any calls which use more sophisticated services than traditional call routing by dialed digits. Figure 2.1 below shows the functional entities, their relationships and how these entities are typically grouped into platforms in a network. All control relationships are normally supported by signaling transported over the reliable SS.7 (Signaling System Number 7) packet data network.

Each function plays a role in the completion of calls which require additional processing to be routed through the network. The CCF takes care of the routing of normal calls through the PSTN and detects when an IN session should be initiated, for example by examining dialed digits. For an IN call the CCF gives control to the co-located SSF which starts an INAP (Intelligent Network Application Protocol) session with the SCF to request any special handling instructions for this call. Normally the SCF will then respond with a destination address (number) to which the call should be routed. Many variations on this basic scheme are possible, involving the SCF querying a database (the SDF) or instructing the SSF to connect the call to a recording playback device in a SRF. The SCF may also instruct the SSF on specific billing arrangements to be used when dealing with this call. Supporting entities provide functionality such as service creation (SCEF) and service management (SMF, SMAF).

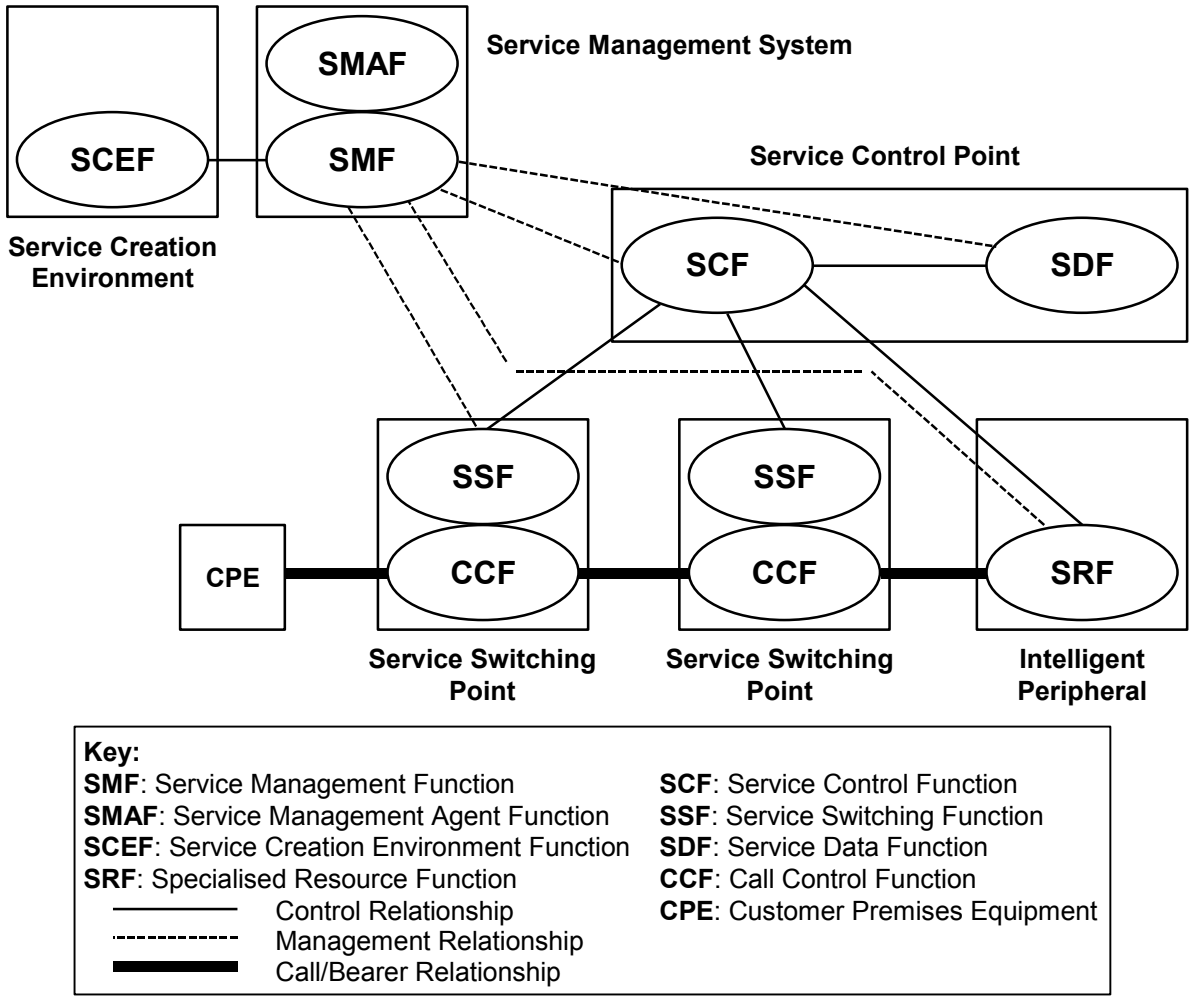


Figure 2.1: The IN Architecture

IN has been structured into a number of modular extensions called Capability Sets [ITUIN]. The first Capability Set (CS-1) was standardised in 1993 and is widely deployed (although often with proprietary extensions). CS-2 has also been standardised and is available from many equipment manufacturers. CS-3 really only exists as a standard at this time and work on CS-4 has begun. Each Capability Set extends the IN model and INAP protocol to deal with more sophisticated services while remaining within the original IN architectural framework.

2.1.2. Driving Forces for IN Evolution

Development of the IN concept has been motivated mainly by:

- a need for a reduction in the time from service conception to deployment in PSTN networks
- a desire to free network operators from dependence on particular equipment vendors
- a wish to enable the provision of services by third party providers

The degree to which existing IN systems meet these goals may be debated, however IN is clearly a commercial reality, with an ever-increasing penetration of IN services and growing demand for more sophisticated features that can be customized by the end-user. Deregulation of telecommunications markets globally will result in a more open environment in which third party service providers will seek access to the public network infrastructure. These trends point towards a growing necessity to overcome some of the constraints of existing IN platforms.

2.1.2.1. Inherent Limitations

The majority of IN implementations are still based on the CS-1 standards, which limit IN service logic to control of (voice) call setup in response to triggers activated at network switches. For some time there has been a clear need to extend the capabilities of IN to allow control of calls throughout their duration and to introduce support for multi-party, multi-connection calls, possibly involving the exchange of multimedia data. It should be noted that ITU-T IN CS-2 does go some way towards realizing this goal. In addition, the manner in which IN services are deployed and accessed should be made more open – for example service logic should be accessible from Internet terminals and contain components residing in both SCPs and Internet nodes. Trigger[1] deployment in switches is also currently a problem area as service deployments can require upgrades of switch software.

2.1.2.2. Service Development and Deployment

As alluded to above, a key motivation for IN was to reduce the time between service conception and full-scale deployment in the public telephony network. Whilst IN has succeeded in cutting this time lag it is evident that even faster service deployment is required today to satisfy rapidly changing customer demands. However the necessary improvement may be hard to achieve in the short term as methodologies and tools for service creation and validation have not yet matured to support the degree of automation required for rapid development. To a large degree this is due to the complexity and time-consuming nature of service validation in general and the difficulties associated with detection of unwanted interactions between services in particular. In addition, service components are generally not developed with re-usability or customisability in mind, thus development times for new services may be unnecessarily lengthy. Finally, generic service deployment and management are still not mature technologies. This means that portability of service logic from one IN platform to another (even within one manufacturer's range of platforms) is very time consuming and uneconomical for all but the most basic services.

2.1.2.3. Vendor/Technology Independence

Stringent reliability and performance targets for telephony services have meant that fault-tolerant

[1] A trigger determines when a call being processed at the SSF/CCF should interact with a SCF.

equipment and specialized software has been necessary for the realization of IN platforms. Variations in versions of the communications protocols and service logic mean that in many cases the task of getting equipment from different vendors to inter-operate properly is very difficult. These factors contribute greatly to the difficulties for smaller vendors and third party service providers who wish to enter the IN market. Many see the implementation of IN elements using middleware solutions such as CORBA as offering the potential to overcome these difficulties by making service logic independent from underlying hardware/software architectures and communications protocols.

2.1.2.4. Openness

A key barrier to the development of IN into a truly open service environment is the lack of standardized interfaces for service creation, management and deployment that can be used by non-specialists to develop IN services using commercial software development methods and tools. These open interfaces may even be tailored for the use of end-users wishing to create/customize services to meet their own needs. An additional limitation is a lack of facilities for brokerage of service components that would promote dynamic re-use of IN software. However, this would require significant enhancements to current IN equipment which, as discussed above, tends to be very proprietary in nature.

2.1.2.5. Integration of Services for Packet Networks

Currently IN is focussed on delivering services for the PSTN (Public Switched Telephone Network) and more recently the PLMN (Public Land Mobile Network). Operators and customers realise that new services delivered by packet networks (such as the Internet) which are integrated with the PSTN and PLMN represent vast potential for growth. From the consumers point of view, standard telephony services (which are currently provided by IN) should be available in the new domain. From the operators point of view, the currently installed base of IN equipment should be leveraged to provide these services. The current incompatibility of the closed telephony network and the open environment of the Internet must be addressed.

Consideration of the limitations discussed above leads to the view that the key requirements for future development of IN are that customers can access a wider variety of services in a wider variety of ways and that they can customize existing services or even create new ones as simply, quickly and cost effectively as possible. This requires operators to have more efficient and open service creation, deployment and management facilities that are integrated into IN service platforms.

2.1.3. Common Attributes of IN Evolution Paths

Although it is impossible to predict exactly how the current standardization efforts will impact the Intelligent Network of the future, there are some overall technological and business trends. This section

attempts to identify these key trends that will have a far reaching impact on service providers, network operators and service subscribers.

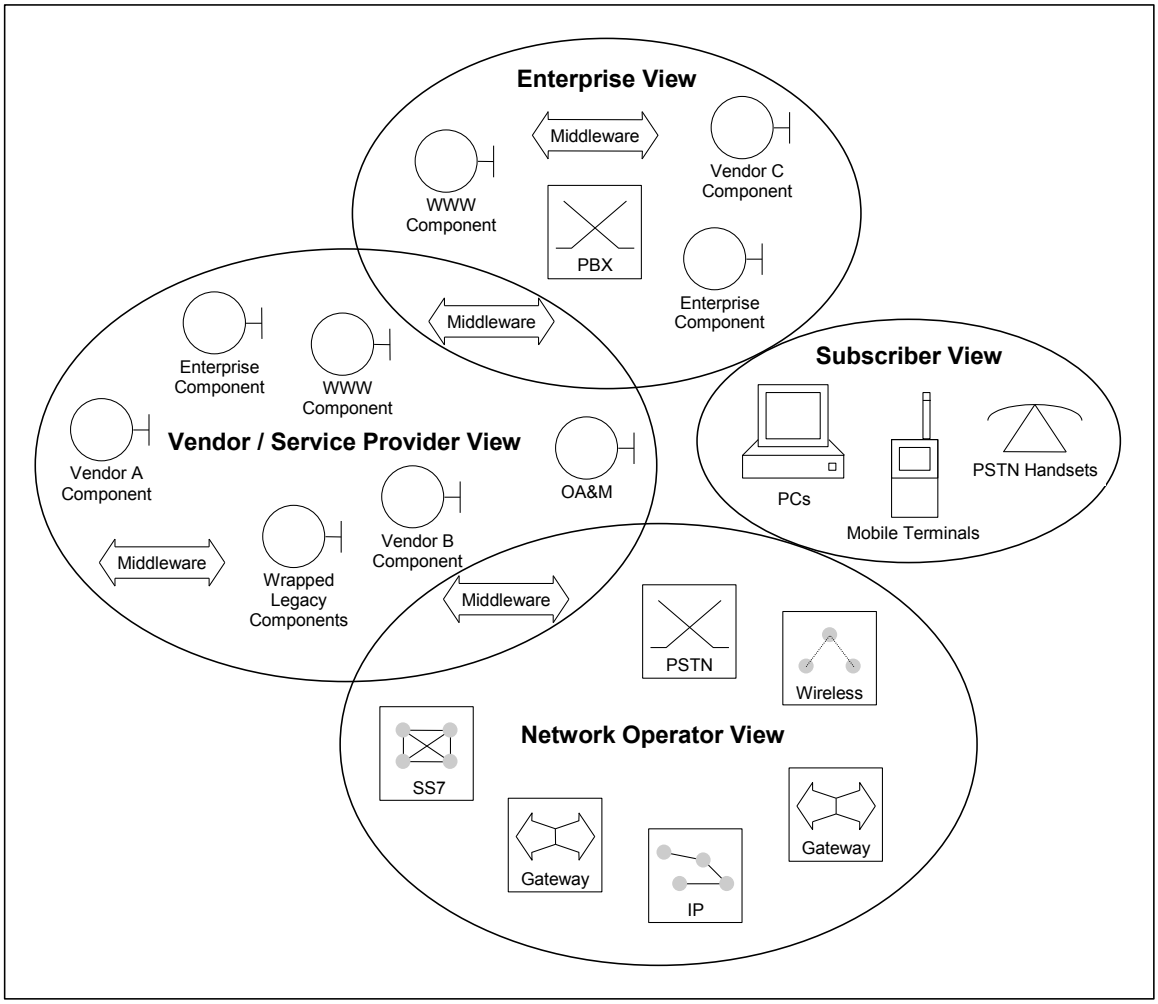
From the view point of the emerging standards, one of the key features in current thinking is the use of generic middleware technologies such as CORBA. Middleware will be used for the provisioning of distributed service logic in an open-market environment and also for providing a platform for leverage of resources, such as the Internet, which are largely isolated from the current Intelligent Network. Conversely the use of middleware and generic protocols also opens the network to access to and from other domains. The trend of standardizing fine grained, middleware interfaces allows for the possibility of smaller, more manageable standard software components making up an overall service architecture. This in turn allows for a more open market place with opportunities for many independent service vendors and third party service providers, whose presence makes it easier for operators to minimize technology lock-ins.

An alternative point of view sees simple, extensible IETF and W3C (World Wide Web Consortium) protocols and standards as the common glue that bind these domains together. It may be argued that the sum of these technologies, for example XML and HTTP as found in the W3C SOAP (simple object access protocol, is equivalent to existing middleware solutions such as CORBA. For this approach the attractions of minimalist standardisation, apparent simplicity, wide availability of implementations and knowledgeable staff must be offset against current architectural holes in the infrastructure, the real complexity of building hybrid systems that combine multiple “simple” architectures and the lack of compatability with traditional telecommunications design methodologies.

Standardization of bridging interfaces is also a common trend, that allows a merging of currently isolated enterprise, public and private service networks such as PBX, Intranet, Internet and existing INs. This will enable more complex and flexible services to be offered. Access by differing terminal equipment will be more easily coordinated and interact to provide the subscriber with more powerful interfaces for service subscription, access and configuration.

A general trend found in current standards work includes the use of generic IT software solutions for acceleration of the development cycle, sophisticated service provisioning, increasing the size of the recruitment pool of knowledgeable staff and lowering costs by avoiding specialist solutions. This has the effect of allowing service execution to be distributed across many domains, allowing reuse of existing service networks in new ways and allowing access to services by different users with different terminals than originally intended.

Figure 2.2 describes pictorially how, based on the technologies discussed here, different elements of existing and future technologies may combine to form the future network as seen from the service provider, network operator and subscriber viewpoints.



Subscriber View

- Multiple, cooperating terminal types
- Service access from any terminal type
- Terminal mobility
- Personal mobility
- Web-based service usage/configuration
- More powerful/useful services

Network Operator View

- Merged network types via gateways
- Coexisting new and legacy technologies/equipment
- Integration of enterprise systems
- Larger choice of equipment vendors
- Distributed service execution and management
- Larger range of service offerings

Enterprise View

- Interworking between network and enterprise technologies/equipment
- Specialisation/Customisation of enterprise systems
- Web-based service usage/configuration
- More powerful/useful services

Vendors / Service Providers View

- Many vendors in more open market
- Smaller service components
- Integration with web-based application servers
- Re-use of existing service platforms
- Use of generic IT hardware/software
- Larger range of product offerings

Figure 2.2: Common Attributes of IN Evolution Paths

2.1.4. IN Evolution Challenges

The future direction of IN development is uncertain, as even the subset of possible approaches discussed so far suggests. As there are common themes across the technologies, there are common deficiencies to be contemplated. In this section some of the more pressing issues are discussed.

2.1.4.1. Difficulty of Specification

A common pitfall for evolution technologies is the specification of an enhanced service architecture which is dependent upon other technologies which are not yet mature enough to be deployed (and hence may never be deployed due to other market forces). All of our chosen technologies are specified by IT or communications technology fora that depend upon rapid implementation and wide-ranging industrial participation for specification adoption. However only IETF PINT is unhindered by dependencies on middleware technologies which may delay deployment of either Parlay or OMG IN/CORBA Interworking solutions. Of course IETF PINT has a much more limited scope and this will also be a key to its rapid implementation and deployment. Technical difficulties with any proposed change in the existing IN architecture will not always be immediately apparent from specifications – actual implementation experience is always required.

2.1.4.2. Non-Functional Requirements of Telecommunications Systems

Performance and reliability will always be key attributes of telecommunications systems that are not traditionally present in enterprise IT systems. There are several facets of the technologies presented in this paper which are impacted by this. Service logic distribution, a common attribute of all of these systems, involves a communications and processing cost which will have to be justified in terms of the added capabilities or other savings. TCP/IP networks are generally envisaged as the transport for the protocols supporting the new technologies and it is well known that they are currently less well suited to telecommunications demands in terms of predictability, congestion avoidance and reliable fault recovery than the well established SS.7 family of protocols used for traditional signalling applications. Finally middleware solutions have not yet proved themselves capable of the high availability requirements (minutes downtime per year) of the telecommunications service network.

2.1.4.3. Fragmentation of Standardisation Activity

Maintaining and expanding network elements based on one or more families of network protocols has always been a major effort for operators and vendors. We now have a situation where the number of design ideologies is expanding from the traditional telecommunications (e.g. ITU-T) view to include data communications (IETF) and distributed object technology. This brings with it differing approaches to protocol design that may have to be accommodated within one product. This will increase training costs for staff and reduce the number of common software elements in a network element. Of course the overall vision for evolution will also be split amongst differing bodies with the likely result of a gradual decline in the size of evolutionary steps. Hopefully this will lead to a more dynamic environment where gradual change is more normal and advances may more easily be introduced into the network. Alternatively more and more processing power, bandwidth and development effort may be devoted to maintaining backwards compatibility and interworking between solutions.

2.1.4.4. Convergence with IP Solutions

Every standardisation effort discussed in this article includes, at least in some form, an integration of IN and IP solutions. As can be seen from even this limited selection, there are many ways in which this convergence can be achieved. The reality is that most approaches will not survive in the long term, one or more clear winners are likely to emerge. It is even possible that pervasive IN-MoIP integration will not occur. There are many in the internet community who believe that throwing off the shackles of traditional telecommunications systems such as IN are part of the new departure offered by protocols such as SIP. For them, to bind MoIP to IN would stifle the potential for unforeseen service developments in the MoIP world. Thus it is more likely that centralised solutions such as IN will be the servants rather than the masters of future IP terminals. The reality is that at least some level of IN/MoIP integration is made necessary by the large installed base of traditional telephony equipment.

2.1.4.5. Opening the IN Services Plane

Traditionally IN SCP systems have had to deal with service requests originating from SSPs across a well engineered SS.7 network - a relatively well controlled environment. All of the architectures featured in this article include the possibility of new types of service request arriving through other channels, probably carried on an IP network. If this is coupled with user service customization traffic arriving via a WAP gateway (an application with huge growth potential) quite a different operating environment is created for service execution nodes. How will this traffic be managed? What mechanisms will be used to prioritize different types of requests? In order to answer the questions and others, the necessary architectural elements will have to be put in place before these systems are viable in an operational environment. Undoubtedly proprietary solutions will be available first, but this development probably requires some rethinking of the current IN model and surely some standardisation will be necessary to provide seamless inter-domain solutions.

2.1.4.6. A Brave New World

Finally, the very act of opening up operator networks will have a profound effect on issues such as security, dimensioning, network planning, fault management and network integrity. Traditionally closed worlds, operator networks will be forced to evolve to include complex gateway functions which both protect the network and monitor external usage for legal and accounting purposes. Although similar situations can occur today in the case of inter-provider agreements, the scale of the problem will be enormously magnified if and when any SME (Small to Medium Enterprise) or individual can have such an arrangement with a provider. Large-scale issues of network strategy are also looming. Many operators are currently both transport and service providers. Is this the best way to sustain growth? Ease of IT integration and the proliferation of data network access for individuals will force operators to provide a more dynamic environment. This interconnected world will provide challenges for today's

hierarchical management systems and thus requires new solutions to be in place.

2.2. CORBA

The Common Object Request Broker Architecture (CORBA) is a set of middleware system standards [CORBA] and application standards developed by the Object Management Group (OMG) industry consortium. Middleware represents an approach to simplifying the development and integration of network-based software distributed across heterogeneous computing nodes. The middleware layer exists above the heterogeneous operating systems on each node and the communications protocols used to link the nodes. This provides a uniform platform on which the distributed applications can execute and interact. Key design issues to be addressed in a distributed environment are:

- **Model**: What are the entities that comprise the distributed system? How do they interoperate? How is their behaviour specified?
- **Architecture**: How are the distributed entities named, located and protected?
- **Engineering**: How is performance achieved? Is the distribution of the system transparent to the system programmer or just the user of the system?

Early middleware systems were based on distributed or remote Inter Process Communication (IPC) mechanisms and a Client-Server model [Amoeba]. These systems are based on a system model that utilises units of distribution such as modules of code or abstract data types and generally mix both architectural and engineering design concerns in a single specification such as a communication protocol definition. More recently initiatives like the ISO Open Distributed Processing [ODP] and CORBA have more generalised architectural models based on distributed objects, which are themselves a generalisation of the client-server model. CORBA is designed to allow components (groups of collaborating objects) to discover each other and interoperate on an object bus (the Object Request Broker, or ORB). In addition it specifies a set of bus-related services for objects (Naming, Life Cycle, Transactions, Events, Security, etc.). CORBA objects are defined by their interfaces written in OMG-Interface Definition Language (OMG-IDL) and bindings between all common programming languages and IDL are defined. This allows systems to be built on both heterogeneous computing environments and heterogeneous programming environments. The OMG Object Management Architecture as realised by the CORBA environment is illustrated in figure 2.3.

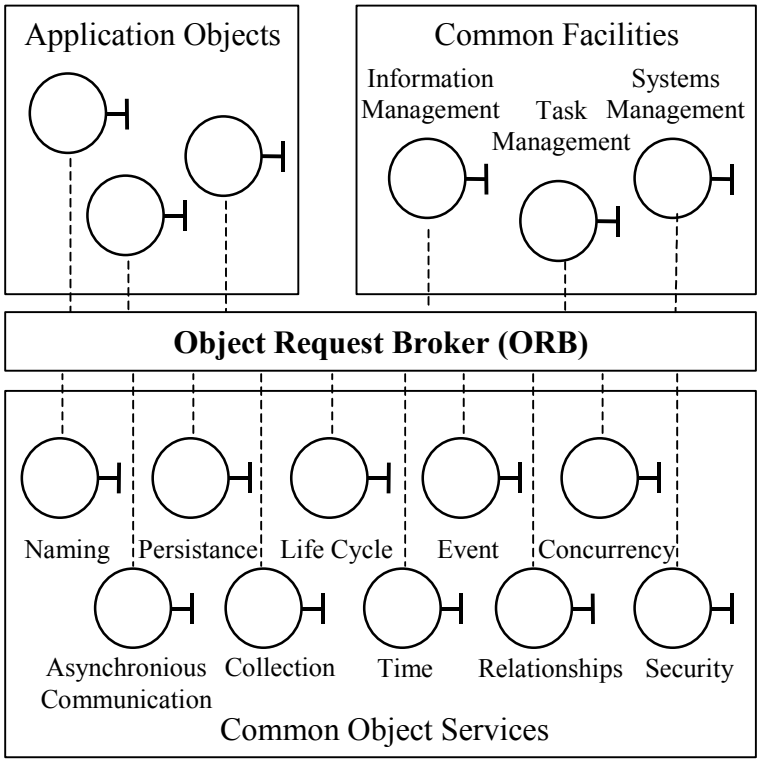


Figure 2.3: OMG Object Management Architecture or CORBA Environment

2.2.1. The CORBA Network Architecture

The CORBA distributed computing environment (DCE) is network protocol neutral. This is achieved by a modular, layered design which enables the incorporation of a multitude of networking technologies. However there are still real limitations placed on a CORBA implementation by the characteristics of the underlying network technology in terms of robustness, message response times and ability to meet real-time constraints. Telecommunications signalling applications are by their very nature distributed real-time applications and have been well served in the past by dedicated SS.7 networks. Any technology which seeks to compete with existing telecommunications service infrastructures must be able to demonstrate similar performance and robustness guarantees to the current SS.7-based solution.

The CORBA DCE is structured to allow the incorporation of the multitude of network technologies available now or in the future [CORBA]. As illustrated in figure 2.4 it is based on classical layered protocol architecture. CORBA applications (servers and clients), reside at the top of the CORBA network architecture, the next layer is the ORB or broker layer providing object location, method invocation and other DCE support services. Below the ORB layer there is the inter-ORB-protocol (IOP) layer which includes the network protocols which provide inter-ORB (or inter-DCE node) communication services.

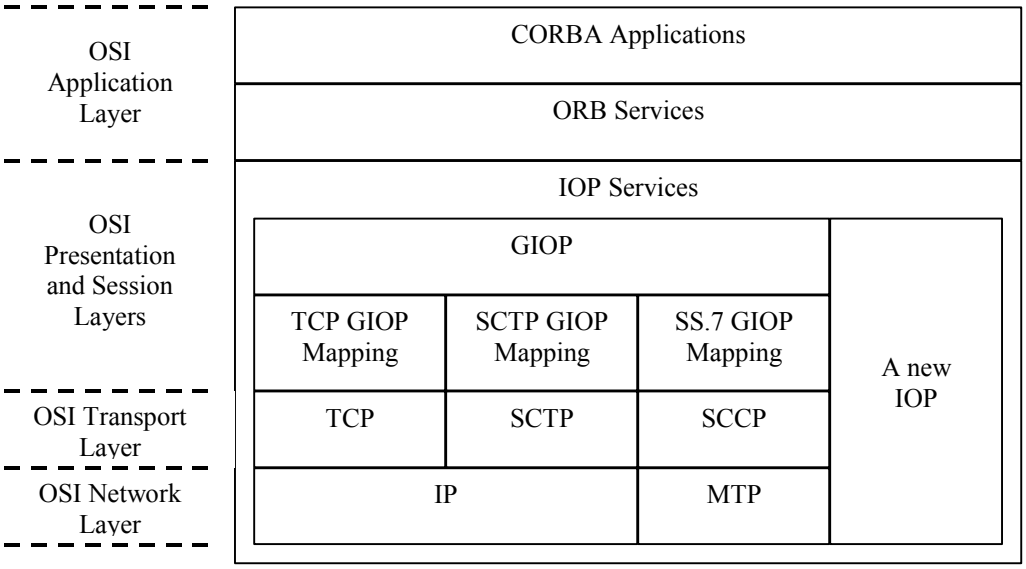


Figure 2.4: CORBA Networking Architecture

Within the IOP layer there is sub-layering to allow re-use of the standard General Inter-Orb Protocol (GIOP) over heterogeneous network protocols. This GIOP provides a standard syntax and semantics for inter-ORB communications but requires certain services from the transport layer protocols and thus may not be suitable for use with all network protocols. For a particular transport protocol to be used with GIOP a thin mapping layer must be developed to map GIOP service requests onto transport protocol services. Alternatively, specialised IOPs called Environment Specific Inter-ORB Protocols (ESIOP) can be created to leverage the services offered by particular transport protocols. Examples of ESIOPs are the DCE-CIOP, designed to allow communication between ORBs in Open Software foundation DCE environments or SECIOP which provides additional specialised ORB security services.

2.2.2. CORBA in the Telecommunications Domain

Middleware technologies like CORBA are increasingly seen as the appropriate infrastructure for future service networks due to the inherent technological advantages brought to bear by distributed, object oriented processing environments. Indeed, CORBA has already been recognised by the ANSI T1 Committee as the basis for defining a framework and generic network information which has been submitted to the ITU-T in an attempt to develop international standards for CORBA-based network management interfaces [T1MGT].

In the same vein, the CORBA interfaces provided by a distributed services application specification would provide standardised interfaces to allow more open and distributed service. Also, the common CORBA approach to management and service provisioning produces a more integrated network and

less cumbersome service management. It would also facilitate increased interconnection capabilities with external resources such as the Internet and private databases. This approach has the added advantage of providing a homogeneous interface for any SS.7 protocol stack implementation. Currently most SS.7 stacks have a proprietary, vendor-supplied API. This independence reduces technology lock-ins, allowing service creation that is independent of proprietary SS.7 protocol stack implementations. OMG standards also have the desirable feature of fast standardisation and short time to market. Leverage of existing, off-the-shelf CORBA Services, such as security, naming, messaging, notification, could also help to accelerate application development.

CORBA still has shortcomings when expected to operate in a highly fault intolerant, real-time environment as is expected of telecommunications systems. From a more general viewpoint since CORBA standardization is controlled by an IT industry body rather than a telecommunications body, it may be difficult to impose telecommunications systems requirements on standards which are based on an architecture for implementation of more general purpose distributed systems. However, since the foundation of the OMG, the Telecommunications Domain Task Force (TelDTF) has contributed expertise and key specifications such as the Notification service (enhanced events used in telecoms management systems) to the CORBA architecture. The CORBA log and file transfer services were also produced within the TelDTF with the explicit purpose of allowing standard telecommunications billing and performance data to be collected across an ORB. The realtime Platform Special Interest Group within the OMG has started to publish a series of standards which will eventually allow the purchase of off the shelf ORB products which are suitable for realtime tasks. Currently a number of specialised proprietary ORB solutions are available in this application space.

2.2.3. CORBA for Telecommunications Services Research

For many years the main driver of middleware-based telecommunications services research was the TINA Consortium (TINA-C). The new model [TINA] for telecommunications service delivery proposed by the TINA-C was inherently object-oriented, distributed and dependent on a middleware component called a Distributed Processing Environment (DPE). This DPE became increasingly identified with a CORBA ORB and many prototype TINA implementations were based on CORBA. There is a huge body of publications documenting the analysis and implementation of CORBA-based TINA service architectures. This work spans service creation, service/DPE performance, prototyping, POTS integration, IN integration and a host of other areas, see for example [TINA97]. Unfortunately the TINA architecture had a number of serious limitations, for example it had no clear method of integration with the existing IN service infrastructure, the network architecture was heavily ATM-based and the service architecture had difficulty encapsulating the end-to-end nature Internet communications. The TINA-C ceased operation in 2000 but many aspects of their specification work has either been

adopted by or influenced the work of standards consortia such as the OMG, 3GPP, TMF and ITU-T.

In parallel with the TINA work there were a number of non-TINA based initiatives to look at CORBA-based services and signalling in general. Eurescom set up project P508 to look at evolution or migration paths to TINA [P508]. This highlighted the need for operators to continue providing IN services and the strong business case for IN inter-working as a key feature of any future solution. Two technical solutions are presented to the issue of IN integration, first the idea that SS.7 could be used as a transport for middleware/CORBA messages and secondly a basic outline of the required capabilities of an IN-CORBA gateway. Their final major finding was the importance of the continued development of the ORB specifications in order to support real-time applications.

During 1996 the OMG TelDTF began to draft a white paper on CORBA-based IN which was important because it solicited responses from industry on their internal CORBA-based IN prototyping experiences. Most of this work had been carried out in either TINA or TINA-IN integration context but its main importance was due to the fact that it brought into the public domain work that had been carried on internally by a number of equipment vendors and operators, notably AT&T [MAZ, MAZ2] and Alcatel [AL97]. The AT&T work by Mazumdar and Mitra was significant because it refined the gateway model presented in the Eurescom work by describing how OSI ROS [X880] definitions could be mapped to OMG IDL with the same mapping for basic ASN.1 types [X680] previously defined by the X/Open Joint Inter-Domain Management group [JIDM]. It also outlined some other possibilities for inter-working scenarios and re-use of standard CORBA services to support the gateway or native CORBA IN objects. The Alcatel work added some background on performance testing of their internal prototypes and yet more possible inter-working scenarios. This work was then all collated into an OMG white paper on CORBA-based IN [OMGWP]. The white paper posed as many technical questions as it answered but it provided a strong list of technical motivations for the development of CORBA-based IN solutions. These included:

- The natural role of CORBA-based middleware as a common glue to bind together the multi-protocol, multi-vendor, multi-service type telecommunications network environment. This heterogeneity has vastly increased the complexity of software design for this environment. CORBA naturally encapsulates this complexity in a common, extensible architecture.
- The potential to merge the historically separate telecommunications management and control (signalling) domains. This would have resultant productivity gains in service creation, service deployment, IT integration with telecommunications services and the already required integration of some management functions during service execution and vice-versa.

After drafting the CORBA based IN white paper, the OMG issued an RFP (Request for Proposals) to start a CORBA-based IN standardisation initiative. This initial RFP was responded to by Iona Technologies, AT&T, GMD Fokus, Nortel, and Teltec DCU. This standardisation process resulted in

the publication of the OMG CORBA/TC Interworking and SCCP Inter-ORB Protocol Specification [OMGIN]. This work is discussed further in Chapter 4.

In addition to work within the OMG on CORBA-based telecommunications services the Xbind consortium created a lightweight architecture for CORBA-based ATM switch control [Laz95]. This included a multimedia service architecture [Laz96] that had parallels with both TINA and the work carried out in the OMG. In common with the TINA architecture, it focused on next generation ATM-based service development rather than integration with the existing telecommunications infrastructure. This work was eventually commercialised by Xbind Inc. but was unsuccessful in the marketplace. The OPENSIG (Open Signalling) group has continued this work with a new focus on non-ATM-based solutions. Much of this work is closer to the network elements (network programming) than the OMG work that instead provides a framework for the integration of traditional telephony signalling protocols into the CORBA domain. The OMG work leaves almost total flexibility for the precise structure of the systems in the CORBA domain.

2.3. IP-based Signalling Transport (SCTP)

A key requirement for any future service architecture is how well it can integrate into the increasingly prevalent IP network infrastructure. This is for two main reasons:

- Users have become familiar with the services offered by the Internet and their corporate Intranets. These are increasingly seen as ubiquitous and interconnectivity with them is now a fundamental requirement.
- From a service provider point of view it is desirable to leverage the massive development effort that has gone into producing off the shelf IP network technology to reduce costs.

However, the integration of IP technology into the telecommunications service control network potentially has many risks due to the differing requirements of Internet users and real-time network control traffic, the relative immaturity of teletraffic engineering methods and tools for IP networks and uncertainties about the nature of TCP/IP traffic.

In this document we concentrate on the behaviour of SCTP at the transport layer rather than dwelling on network protocol issues such as IP, DiffServ or RSVP. Instead we just assume that the network layer provides a basic unreliable datagram transfer service. In this scheme, the addition of priority flows to the network can be modelled as reduced or increased resources for the signalling traffic that we are interested in. As we will see in Chapter 6, our results can be used to define Differv or RSVP admission control policies.

2.3.1. SCTP Protocol Architecture

For many years SS.7 has been the dominant bearer of signalling traffic for telecommunications networks, but recently many proprietary solutions for transporting signalling traffic over IP have appeared. This approach promises tighter integration with VoIP solutions and ultimately the possibility of a common core network transporting both signalling and media traffic. Driven by industry interest, and general agreement on the unsuitability of either TCP or UDP, the IETF Signalling Transport (SIGTRAN) group was formed in 1999 to standardize a suitable transport protocol for signalling traffic over IP. Stream Control Transmission Protocol (SCTP) is the result of this work, recently published as RFC 2960 [SX00] by The Internet Society.

SCTP is characterized in a number of current Internet drafts that discuss its capabilities and applicability to both the signalling domain [Coe01] and more general applications [Coe02]. Essentially, SCTP is a reliable, message-oriented data transport protocol that supports multiple streams within an association and hosts with multiple network addresses (called multi-homing in SCTP). All of these properties were chosen to make SCTP more suitable for signalling transport than TCP. For a more complete overview of SCTP see Jungmaier *et al.* [JST00].

SCTP congestion control mechanisms are based upon TCP congestion control principals [APS99], and the mechanisms are based on the optional SACK extensions to TCP [MMFR96]. Modern TCP congestion control principals originate with Van Jacobson in 1988 [Jac88] and they have been refined and investigated by a number of researchers [for example MM96a, All98, APS99, HPF99] since then. Although many of the leading TCP congestion control researchers participated in the specification of SCTP, to our knowledge no work has yet appeared that specifically deals with SCTP congestion control, although some useful performance comparisons with TCP were included in work carried out in 2000 by Jungmaier *et al.* [JST00].

2.3.2. SCTP Congestion Control

The basis of SCTP congestion control is an amalgamation of current best practice for TCP implementations with extensions to deal with the multi-homing aspect of SCTP and modifications due to the message rather than stream-based nature of the protocol. The standard specifies an adaptive sliding window control with versions of the well known TCP slow-start, congestion avoidance, fast retransmit and fast recovery mechanisms. In addition recent work on TCP, such as congestion window validation [HPF99], is also incorporated. One currently optional mechanism for TCP, the use of selective acknowledgements (SACKs) [MMFR96] to report out of sequence data arriving at the receiver, has been incorporated into SCTP as the baseline for congestion control implementation. This is due to the demonstrated superiority [FF96] of this mechanism to earlier TCP congestion control

options. Although the possibility to support IP Explicit Congestion Notification (ECN) [RF99] has been incorporated into SCTP, this mechanism is optional and (as with TCP) a packet loss is the normal method of congestion indication.

2.3.2.1. SCTP Associations

Data transfer between two hosts in SCTP takes place in the context of an association. All transfer of data between hosts is encapsulated in SCTP datagrams, which contain a common header and a sequence of structures called chunks. These chunks may be either SCTP control information, such as a SACK chunk, or user data carried in a data chunk. Within the association data chunks belong to a specific stream of data. Normally within a stream data is ordered, although transfer of unordered data is also supported, and ordered data that has arrived out of sequence is buffered until the missing data has arrived.

During association initialization the SCTP endpoints exchange the size of their receiver window (an indication of the space available in their inbound buffer) and the initial transmission sequence number (TSN) of the user data chunks[2] to be exchanged during the association. All data chunks transmitted receive a unique TSN from a 32-bit namespace. Ordered data within a stream also receives a unique stream sequence number (SSN) from a 16-bit namespace. User data is not limited in size as it may be fragmented over multiple data chunks. Individual data chunks may be up to 655634 bytes in length. SCTP should ensure that each SCTP datagram including any chunks fits within one IP datagram and fragments user data into multiple data chunks as necessary.

2.3.2.2. Monitoring Data Delivery

In order to track the amount of data currently within the network the SCTP sender uses the TSN and the byte size of the corresponding data chunk. Data chunks arriving at the SCTP receiver are acknowledged by transmitting a SCTP packet with a SACK control chunk. The SACK chunk carries several pieces of information:

- The Cumulative TSN Ack, which records the highest sequential TSN received.
- The Advertised Receiver Window Credit, which reflects the current capacity of the receiver's inbound buffer.
- A sequence of Gap Ack Blocks, which record any out of sequence TSNs received.
- A sequence of Duplicate TSNs, which record any TSNs for which duplicates have been received.

Data is not considered fully delivered until the cumulative TSN Ack point advances past its TSN. The

[2] Each of these forms a distinct user-generated message to be carried by SCTP unlike TCP's stream interface

information in the Gap Ack Blocks corresponds to TCP SACK blocks. Under normal conditions SCTP uses a delayed acknowledgement scheme that sends one SACK for every second incoming packet that contains one or more new data chunks.

In order to perform congestion control SCTP uses a number of internal variables to control the rate at which data is injected into the network. These are as follows (all are recorded in bytes):

- Receiver window size (*rwnd*) – this corresponds to a sender's view of the receiver's incoming buffer space.
- Congestion control window (*cwnd*) – this corresponds to the sender's view of network conditions. The initial value of *cwnd* is less than or equal to twice the path maximum transmission unit (MTU), corresponding to the more aggressive value recently adopted for TCP in [15].
- Slow-start threshold (*ssthresh*) – the sender uses this to distinguish between slow-start and congestion avoidance phases.
- Partial bytes acked – the sender uses this to limit *cwnd* adjustment to once per RTT (Round-Trip Time) during congestion avoidance phases.

The receiver window size is always kept on a per association basis, in a multi-homing environment the other values must be recorded on a per destination address basis. When multi-homed a SCTP endpoint designates one path as the primary and sends all data along it. Other paths are considered secondary and are only used for retransmission of data marked as lost on the primary path or for handover in the case of failure of the current primary path.

2.3.2.3. Behaviour

Normal operation of SCTP follows TCP in allowing a sender to inject Min (*rwnd*, *cwnd*) data into the network. Although in order to protect against lost SACKs when *rwnd* is zero it is permitted to have one packet in flight (if allowed by *cwnd*) to probe for changes in the advertised receiver window credit.

During slow-start (at the start of an association, after a transmission time-out or after a long idle period) *cwnd* is increased exponentially to probe the network for capacity. This occurs while $cwnd \leq ssthresh$. During this phase *cwnd* may only be increased by Min(total size of new data acknowledged by a SACK, path MTU) and only if the SACK advances the cumulative TSN ack point (i.e. acknowledges new sequential data). This means that duplicate SACKs (ones which do not advance the cumulative TSN Ack point) can only enable new data transmission by reducing the sender's view of how much data is currently in the network (flightsize).

During the congestion avoidance phase ($cwnd > ssthresh$) *cwnd* may be incremented by no more than 1 MTU per RTT if the sender has *cwnd* or more bytes of data outstanding. This produces a linear increase

in *cwnd*, up to the limit of the initial *rwnd* exchanged at association initialization.

Whenever a SCTP sender is not transmitting to a receiver address (both during slow start and congestion avoidance), the *cwnd* is adjusted to Max (*cwnd*/2, 2*MTU) per RTO (Retransmission Timeout value). This protective measure decreases *cwnd* as network conditions are unknown for a destination to which the sender is not transmitting. This is similar to the approach for TCP outlined in [HPF99].

There are two methods of packet loss detection (congestion notification) defined in SCTP:

1. *Detection of gaps in received TSNs through Gap Ack reports in a SACK*. Normally a sender will wait for four consecutive reports before reacting. This will put the sender in slow-start mode with a reduced *cwnd*. This forces a sender to wait before transmitting new data for either the SACK advancing the cumulative TSN ack point beyond the missing packet (Normally retransmitted through Fast Retransmit, see below) or until enough duplicate SACKs arrive to reduce the flightsize below the modified *cwnd*.

2. *Timeout of the retransmission timer*. This will have the effect of putting the sender in slow start mode and assure that no more than one packet is in flight until it receives the SACK advancing the cumulative TSN ack point beyond the TSN of the retransmitted packet.

Whenever a SCTP retransmission occurs, the sender bundles as many as possible of the earliest TSNs marked as missing into a single packet and retransmits them.

2.3.2.4. Differences between SCTP and TCP Congestion Control

In this section we highlight significant differences between SCTP and TCP congestion control mechanisms, especially those that will be illustrated by our simulation studies in the next chapter. For the purposes of this section we treat the standard TCP congestion control mechanisms as those laid out in the most recent RFC defining TCP behaviour [APS99].

There are two changes to the control scheme, which represent the most significant departure from standard TCP congestion control mechanisms:

- The direct dependence on the number of bytes acknowledged rather than the number of ACKs received to increase the congestion window. This is a more sophisticated control mechanism and is similar to byte counting proposals for TCP by Allman [All99].
- Exclusive dependence on selective acknowledgement (SCTP Gap Ack block) messages for reporting in the case of packet losses i.e. no duplicate ACK counting. This should ensure greater resilience to packet loss than pre-SACK TCP implementations.

Minor changes to the control mechanisms include:

- Fast retransmission of packets is controlled by *cwnd*. This reduces the speed with which SCTP reacts to packet loss and hence reduces throughput. See the results section.
- No explicit fast recovery phase is required, as no artificial inflation of *cwnd* is required for throughput to be maintained.
- During the congestion avoidance phase *cwnd* may only be increased if the full *cwnd* is currently being used. This restriction is not present in TCP, however current TCP research [All99] indicates that it is preferable to the "standard" TCP approach.
- An unlimited number of gap ack blocks are allowed in SCTP SACK chunks. TCP has a maximum of three SACK blocks. This should make SCTP more resilient to acknowledgement loss and better able to deal with multiple (non-sequential) packet loss than SACK-TCP.
- The value of *cwnd* decays when packets are no longer being transmitted. This has been recommended for TCP by Handley et al. [HPF99].

2.4. IP Transport-level Network Performance Research

In this section we provide a review of the publicly available research studies on TCP and SCTP performance. This necessarily concentrates on TCP due to the relative immaturity of SCTP-specific research and the fact that TCP is a direct precursor to SCTP. We focus on the models employed and the insights gained into the operation and performance of the various TCP variants examined.

2.4.1. Approaches to TCP Network Modelling

During the original design or later extension of a signalling network it is necessary to undertake performance modelling and analysis studies to ensure that the required QoS (Quality of Service) parameters stay within the targets mandated by the network service infrastructure requirements. This has led to a wide range of studies concerned with modelling the traditional telecommunications infrastructure. Similarly when constructing or maintaining an IP-based infrastructure such studies must be performed.

Given the heterogeneous nature and popularity of IP-based networks there is a wide body of literature dealing with network QoS, performance and dimensioning issues. Much of this work has concentrated on IP and sub-IP QoS topics such as MPLS, Diffserve, etc. and is not discussed here. Classical (circuit-based) teletraffic methods depend on the Erlang B family of methods to explore the relationship between network capacity, traffic demands and network performance. This in turn provides network planners and managers with sets of dimensioning requirements and operating points respectively. Unfortunately in the IP traffic space even the normal starting point, the description of traffic demands, is still a research issue. IP traffic modelling is complicated by issues such as the discovery of long range

dependence (LRD) and self-similarity in the observed traffic.

The importance of the closed, elastic nature of TCP flows, due to it's use of a closed loop congestion control algorithm, have been shown to dominate IP LRD and self-similarity issues [Arv99]. The actual network performance observed by users or user-applications is critically linked to the behaviour of the transport layer. Even with satisfactory network layer QoS transport issues must be considered to completely engineer the system. Recently a family of analytic models for calculating the expected send rate, delay and loss of TCP traffic have matured and been shown to yield accurate results for a wide range of observed network conditions [Pad00]. These models still deal with the steady-state cases of long-lived TCP flows on a single path but have been extended to cover both short-lived flows [Car00] and networks of flows [Rou01]. See Firoiu *et al.* [Fir02] for a comprehensive overview of analytic modelling of TCP/IP networks.

There are three main approaches used for the estimation of performance of TCP/IP protocols, analytic modelling, event-based simulation and experimentation. The last approach can be further divided into experiments performed on the actual Internet, lab-based experiments on TCP/IP LANs and lab-based experiments that use some sort of WAN emulator to explore Internet-style scenarios. There is a fundamental Internet modelling problem due to the complexity, diversity and rapid pace of change of the Internet environment itself [Pax01]. Despite this fact there is a huge literature focusing on suggested improvements or investigation of particular TCP issues, for examples see [JB88, MM96a], with supporting modelling and experimental results. Despite the proliferation of work in this area there is a growing unease within the Internet research community [Flo02] about the value of much of this work. In addition to the inherent problems for meaningful Internet modelling, Floyd and Kohler identify common problems with inappropriate use of existing models, poor modelling methodology in terms of fully exploring the system being investigated and most fundamentally a lack of a strong basis in real world measurement for many systems. This emphasis on Internet measurement was also present in Veitch's keynote address to the International Teletraffic Congress in 2001 [Vei01].

Traditional teletraffic engineering approaches emphasise analytic modelling as queuing systems have proved a very powerful technique for analysing network performance and generally require comparatively small amounts of development and computational effort. However the difficulties of TCP/IP traffic characterisation, application of queuing models to closed-loop systems and until recently the relative immaturity of TCP models have restricted the popularity of this approach for Internet protocols. The development of a general TCP steady state single link throughput model is described in Padhye *et al.* [Pad00]. The development and application of this model to networks of TCP sources and dimensioning problems can be seen in [Rou01], [Fir01], [Bu01].

Experimental evaluation and discrete event-based simulation are generally the preferred approaches to

model or protocol enhancement validation for Internet protocols. The existence of NS [NS], a common simulation tool for Internet protocols, has helped unify the simulation research effort and enables independent verification of author's results through publication of simulation scripts. It provides a host of standard models for most adopted or developing Internet protocols. There are also limitations to NS as it assumes that only maximum size packets are transmitted which is true for long-lived flows but these are not typical on the actual Internet. The NS TCP model window size and ACK computations are also based on this fixed maximum segment size, which is specified at the start of the connection. There are also dangers inherent in the use of common simulation environment and models as it produces less independent verification of results. It is well known from the examination of actual implementations of Internet protocols [Pax99] that significant different implementations of the same standard can have significant differences in behaviour. This would also be true for a range of protocol model implementations.

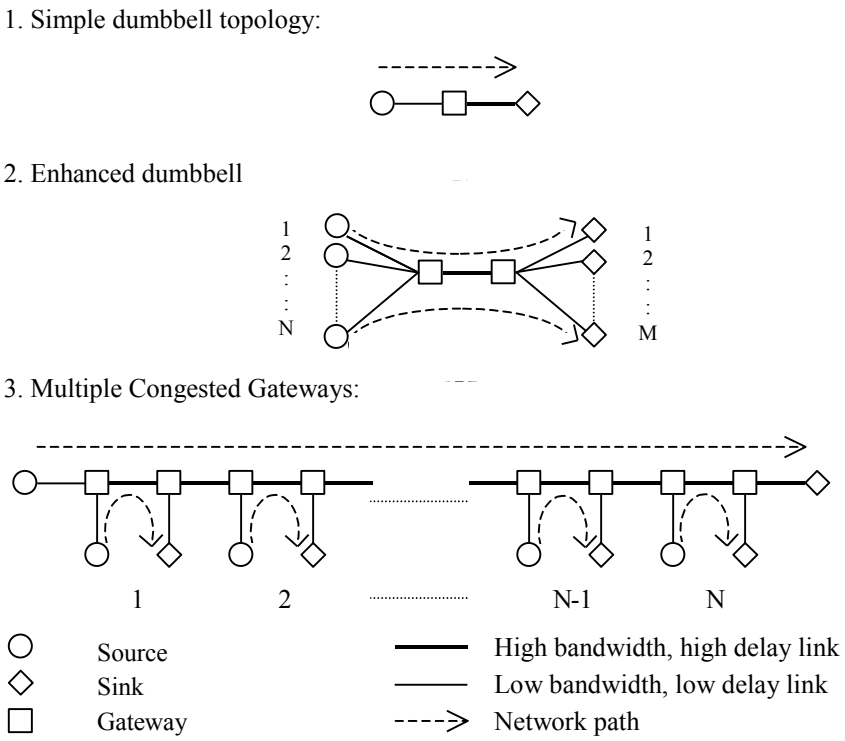


Figure 2.5: Models for simulation-based analysis of TCP.

The experimental approaches used in the protocol evaluation literature are typically based on observations of flows between two widely separated Internet nodes, for an example see [MM96a]. There is a family of Internet protocol measurement tools available for such performance analysis and measurement [Par98]. Generally such experimentation is performed as a final reality check (after event-based simulation) on proposals. There is a powerful argument for such experiments given both the limitations of the NS simulator (see above) and the richness of the environment observed on the

Internet [Pax97a, Pax97b] compared with any simulation. There is also cause for concern about scaling effects and artefacts of the simulation environment impacting on any simulated results [Flo02] in an unpredictable way. Floyd and Kohler go further and say that some protocol development research that has been tested through event-based simulation has sought to solve problems that were present in the simulation but not observed in real systems. They also point out the limitations of experimental approaches when evaluating protocol development due to the impossibility of creating conjectural environments based on posited future developments in Internet size, mix of traffic, types of protocols used, etc.. This is particularly important for protocol development within the IETF as all protocols standardised there must be designed for "big I" Internet use rather than simply for use in restricted or private networks.

When examining TCP congestion control behaviour or evaluating TCP throughput models there are three common network topologies employed. These are illustrated in figure 2.5. The congested "trunk" link is normally characterised as a high bandwidth, high delay (approximately equivalent to a T1) link as might be encountered joining two widely separated Internet sites and the "access" links between the gateways and the traffic sources or sinks have more variable bandwidth and delay properties, but generally with less bandwidth and delay. Most researchers do not vary the characteristics of the links in their network topology. Typically, a range of queue management strategies are modelled in the bottleneck gateway(s); drop-tail, various configurations of RED and perhaps other active queue management techniques. Some papers, especially those primarily dealing with the interaction of TCP controls with active queue management techniques, monitor observed queue lengths in the congested gateways and present results for a range of maximum queue lengths. The NS models of TCP variants are often used to show the difference between various TCP variants (Reno, NewReno, SACK, etc.). The use of NS limits the TCP connections to using one packet size for their entire duration and means that internal TCP congestion control calculations are carried out on a per packet rather than per byte basis. An exception to this is the work of Allman [All98, All99] which is largely based on the idea of extending TCP congestion control algorithms to work on a byte counting rather than packet/segment counting basis thus necessitating extending the functionality of NS and the scope of his experiments. The treatment of background traffic in these studies varies in sophistication from the introduction of small numbers of short-lived, non-adaptive flows at critical points to induce the desired packet loss patterns in the congested link, to extensive models of random background use of the network by variable numbers of other user-protocols with variable use patterns and variable transfer start times during the simulation run. There is a wider variety of network topologies dealt with in the literature covering the application of TCP analytic modelling to network design and traffic engineering, these are specifically described in parallel with those papers in section 2.4.4.

2.4.2. The Evolution of TCP Congestion Controls

Widespread understanding of the key role of TCP in determining network performance dates back to the Internet congestion collapse event in October 1986. As a direct result of this event van Jacobson formulated his congestion control and avoidance scheme for TCP [Jac88] and outlined the lasting congestion control principals for the Internet:

- **Conservation of packets:** For a connection in equilibrium, new segments (packets) should not be injected into the network until an old segment has left. Ideally, this leads to a stable network with a constant number of packets in transit. TCP and SCTP implement this principal by "self clocking" i.e. segment transmissions are generally triggered by returning acknowledgements (ACKs).
- **Congestion avoidance:** For a connection in equilibrium, the transmitter must respond to congestion events by decreasing its rate of transmission and probe for a new equilibrium by gradually increasing its rate of transmission. TCP and SCTP implement this principal by maintaining a congestion window (*cwnd*) state variable representing the amount of outstanding data on the connection i.e. data that has been sent and not yet acknowledged. When a transmitter detects congestion in the network (usually through packet loss) the congestion window is halved. Under other conditions the congestion window is increased linearly by one maximum segment size (MSS) per round trip on the network. Thus, we have an additive increase and multiplicative decrease (AIMD) algorithm.
- **Slow-start:** For a connection yet to reach equilibrium, it must gradually probe the network to determine the equilibrium transmission rate. In TCP and SCTP implementations this is achieved by setting a threshold (*ssthresh*) and increasing the value of *cwnd* by one MSS for each ACK received. This exponential increase continues until *cwnd* reaches the value of the threshold, *ssthresh*, at which point Congestion Avoidance behaviour is invoked. Slow-start is invoked at connection establishment or after retransmission time-outs.

This work was later followed by refinements [Jac90] that added the Fast Recovery and Fast Retransmit algorithms which are intended to preserve self clocking during recovery from a lost segment. The Fast Retransmit algorithm specifies that when three duplicate acknowledgements (i.e. acknowledgements with the same maximum received segment number) are detected that TCP/SCTP must assume the segment has been lost and retransmit it. This is known as a Triple Duplicate ACK or TD event. The Fast Recovery algorithm attempts to estimate how much data remains outstanding in the network by counting duplicate ACKs. It artificially inflates cwnd on each duplicate ACK causing new data to be transmitted. Fast Recovery allows one (halved) window of new data to be transmitted following a Fast Retransmit. Under single packet losses these two algorithms preserve the self-clock and enable TCP/SCTP to keep the network full while recovering from the lost segment. If there are multiple lost segments TCP is unlikely to fully recover, resulting in a timeout and subsequent Slow-start [Flo95]. The first version of TCP to implement all of these features was Reno-TCP that is still one of the most

common versions of TCP seen on the Internet. It is often used as a baseline for compatibility or performance testing.

The next major innovation in TCP congestion control that is widely deployed is selective acknowledgement (SACK) TCP [MMFR96] that is a refined version of the original SACK proposal by Jacobson and Braden [JB88]. This extension uses the extension header fields in the standard TCP ACK message to enable receivers to inform transmitters about which segments have been received when non-contiguous data is received. This is an improvement over Reno TCP as it can only signal the sequence number of the highest contiguous data received. However, space for acknowledging non- contiguous data is limited to two or three segments. The precise algorithm used to exploit this additional information is not standardised but a number of researchers have proposed and validated commonly accepted approaches [MM96a, MM96b, FF96]. The major attribute of SACK TCP is to increase TCP's resilience to multiple packet losses by preserving the self-clock mechanism when confronted by multiple losses within a single congestion window. This in turn reduces the frequency of packet recovery via retransmission timer (RTO) expiry and thus decreases the overall "burstiness" of TCP and increases average throughput [MM96a, MM96b, FF96].

Finally, there is an important body of current TCP research governing TCP behaviour that has recently been standardised but is not yet widely deployed. Two important work areas have been on congestion window validation and byte counting techniques. The work by Handley *et al.* [HPF99] on congestion window validation addresses the issue of defining a suitable decay for the congestion window during application limited periods. They show how *cwnd* (an estimate of network capacity) becomes invalid if network conditions are not frequently sampled, e.g. during application-limited periods, and hence must be adjusted to prevent overloading the network with data. Their control algorithm reduces TCP "burstiness" and thus protects the network by reducing the value of *cwnd* when it is unused. Byte counting techniques [All98] remove some of the assumptions of TCP control algorithms by counting the actual amount of data currently in the network rather than estimating it from the number of ACKs received. This provides better throughput [All98], as TCP is more aware of the true state of the network.

2.4.3. The Development of Single Path Steady State TCP Throughput Models

One of the fundamental results in TCP/IP modelling has been the establishment of the $1/p$ or PFTK (Padhyre, Fireoiu, Townsley and Kurose) family of models that characterise the steady state throughput for a bulk transfer TCP flow as a function of the loss rate and round trip time of the network path traversed. In all cases the average window size (and hence throughput) of a TCP connection through a congested gateway is shown to be proportional to the inverse square root of the packet loss probability on the route. These results all assume that packet losses are independent, slow-start has little effect on

throughput (i.e. the connection spends the majority of it's time in congestion avoidance) and that the sources always have data to send (i.e. are persistent or greedy).

The exact origins of the fundamental ideas behind this model are hard to attribute, as it existed as an informal approximation long before it was formalised. It is probably the case that a number of researchers independently derived the informal version of the model. We can motivate how the $1/p$ law arises as follows:

- Assume that packet losses are independent with small (the exact magnitude is not well defined) probability p and measure time in units of the round trip time of the connection (RTT).
- During the nth RTT the congestion window size (*cwnd*) is defined to be w_n in packets of the Maximum Segment Size (MSS) for the path.
- Thus the probability of one packet loss in that RTT is approximately pw_n and the probability of two or more losses is negligible.
- In the case of only one packet loss it is reasonable to assume that loss will be indicated by the reception of a triple duplicate ACK and the Fast Retransmit/Fast Recovery mechanism will be triggered, causing the retransmission of the packet and a halving of the congestion window size.
- If no loss occurs then the system is assumed to be in congestion avoidance and thus the window will increase by one MSS each RTT. The case of two or more losses is not considered, note that this assumption is more valid for SACK TCP than the older Reno TCP as the SACK option allows multiple packet losses within a window to be recovered from as a single congestion event thus not requiring a retransmission timeout to recover. Therefore:

$$w_n = \begin{cases} w_n + 1, & with\ probability\ 1 - pw_n \\ w_n / 2, & with\ probability\ pw_n \end{cases} \tag{2.1}$$

Taking expectations and neglecting the term $-p\overline{w_n}$ for small p results in

$$\overline{w_{n+1}} \cong \overline{w_n} + 1 - \frac{p\overline{w_n}^2}{2} \tag{2.2}$$

In equilibrium $\overline{w_{n+1}} = \overline{w_n} = \overline{w}$ and therefore a rearrangement of (2.2) gives

$$\overline{w} = \sqrt{\frac{2}{p}} \tag{2.3}$$

It must be noted that (2.3) only holds for light loss rates though light is not well defined.

There is a second common case, periodic loss, loss of every $(1/p)^{th}$ packet. This case will occur on a

congested link as the source probes for more bandwidth and gradually opens its congestion window until it encounters a loss (congestion indication) which halves the congestion window and once again the source probes for more bandwidth. If the path is congested on long time scales (compared with the probing mechanism) then this behaviour will be periodic. Figure 2.6 shows this case with a peak window of *w* and therefore a period of *w*/2 RTTs and immediate rather than delayed acknowledgement.

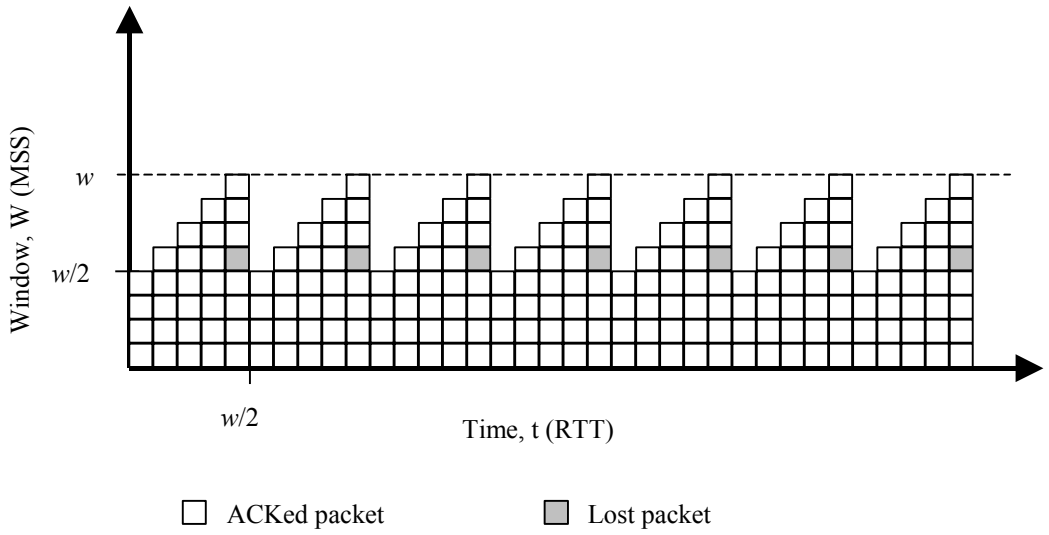


Figure 2.6: Periodic Loss Model.

The area under the graph for one cycle gives the number of packets transmitted in one loss cycle. The area of the region is just the square $(w/2)^2$ plus the triangle $(w/2)^2/2$. Thus each cycle consists of $\left(\frac{w}{2}\right)+\frac{1}{2}\left(\frac{w}{2}\right)^2$ packets and exactly one packet is lost, therefore

$$\left(\frac{w}{2}\right)+\frac{1}{2}\left(\frac{w}{2}\right)^2=\frac{1}{p} \qquad (2.4)$$

Where *p* is the loss rate. The average window size is ¾ of the peak window size *w*, hence

$$\bar{w}=\frac{3}{4}w=\frac{3}{4}\sqrt{\frac{8}{3p}}=\sqrt{\frac{3}{2p}} \qquad (2.5)$$

We can then deduce the throughput (network load) from the average window size from

$$throughput=\frac{data\ per\ cycle}{time\ per\ cycle}=\frac{\bar{w}\times MSS}{RTT} \qquad (2.6)$$

This value for throughput includes retransmissions and is a measure of the data injected into the network. From an application point of view we may wish to determine the goodput which is the rate at

which user data gets through. When the loss rate is low we can approximate this as

$$goodput = (1-p) \times throughput \tag{2.7}$$

Lakshman and Madhow have presented a series of papers on this topic [LM94, LM95a LM95b, LM97] that consider the case of high-bandwidth delay product compared with queuing delays. They assume that random losses are due to fluctuations in non-adaptive, non-TCP flows (perhaps realtime traffic). They note that due to (2.6) TCP is unfair (gives a smaller share of bandwidth) to connections with longer RTTs and as the window increase rate is also dependent on RTT there could be a further bias against long RTT connections. Their analysis concentrates on TCP Tahoe and Reno with almost full flow control including slow start but then approximate away all details to get the simplified result

$w_{ss} = \sqrt{\frac{2}{3p}}$, where w_{ss} is the average slow-start threshold. (2.8)

They concentrate on this as a means of determining when the throughput is limited by the flow control or by the capacity of the network. Thus they never derive the $1/\sqrt{p}$ law for the average throughput but are very close.

The first publication of the $1/\sqrt{p}$ law seems to have been in Ott et. al. [OKM96]. This paper is cautious about the general applicability of their results and specifically name SACK TCP as an idealised case that is best treated by their approach. They note that real losses are correlated rather than independent but hope that RED AQM will improve this. They derive the following expressions for the average window size:

$$\overline{w} = \begin{cases} \frac{1.309}{\sqrt{p}}, & for\ random loss \\ \frac{1.707}{\sqrt{p}}, & for\ periodic loss \end{cases} \tag{2.9}$$

In addition to their formal derivation of the law, it is important that they have shown that it holds for periodic as well as random loss as this confirms that it does not rely on purely random losses. A limitation of their work is that most of the results rely on not having bursts of losses, which is certainly observed in real networks. They also present some preliminary work on the distributions of window sizes. Finally, their work extends the basic outline given above by accounting for delayed acknowledgements that are likely to be used in a modern TCP deployment.

Floyd [Flo91] does not consider the $1/\sqrt{p}$ law directly but considers the effect on multiple TCP connections through multiple gateways (as shown in figure 2.7). A bias is shown against connections

that pass through multiple congested gateways. Most of the work in this paper considers the effects of various queue management mechanisms and their interaction with different congestion control behaviours (constant rate or additive increase). Rather than simple independent losses this paper assumes independent Poisson processes at each gateway generate them. The loss rate on a particular connection is assumed to be proportional to the connection's current throughput on a given link. This assumption is supported by simulation results for RED gateways. The multiple bottleneck connection is assumed to have a percentage throughput q and therefore a probability of loss occurring on this connection of q. There are several claims/theorems stated. The most important for our purposes is Floyd's corollary 3 which concerns the loss rate observed by a TCP congestion avoidance-like linear rate window size increase control. For a single bottleneck with two connections with different RTTs, $r_0 \geq r_1$, and linear rate increase every RTT seconds then

$$q = \frac{1}{1 + \frac{r_0}{r_1}} \qquad (2.10)$$

Note the bias against connections with longer RTTs. Floyd's claim 5 generalises this to allow for multiple congested gateways with differing rate control mechanisms and Floyd's corollary 6 restricts this result for multiple competing TCP flows (i.e. all flows in congestion avoidance or linear increase). This most useful result is as follows:

$$q = \frac{1}{1 + \sqrt{n}(2n-1)} \qquad (2.11)$$

Finally Floyd's corollary 7 considers the case where the loss rate is not the percentage throughput but some function of it. Each of these claims and corollaries is presented with a proof but these are superseded by the proofs of the $1/\sqrt{p}$ law (above). Floyd's results are particularly useful when we begin to consider network models rather than modelling a single path or link.

The next stage in the development of the $1/\sqrt{p}$ law was undertaken by Mathis *et al*.. [MSMO97]. In this paper both a periodic and random result are given (with proof only for periodic loss) and Dave Clark and Matthew Mathis are credited with an earlier approximate result for the $1/\sqrt{p}$ law (through private communication rather than publication). The authors cite a number of limitations of the law:

- receiver windows are not considered
- only persistent sources are treated
- there is no modelling of TCP timeout behaviour
- not applicable to TCP implementations that exhibit go back N behaviour or TCP Vegas

- only equilibrium behaviour is studied.

The most serious of these limitations are the limit to persistent sources and equilibrium behaviour. Many real TCP flows are short-lived [McC00] and hence do not share these characteristics. The authors suggest that all of these limitations except the last will reduce throughput and so we can use their results as an upper bound. They note that the correct metric for p is the number of congestion signals per packet rather than the raw loss rate, this is because TCP variants like SACK will treat multiple lost packets within a single window as a single congestion indication.

Mathis *et al.* verify their results through simulation and TCP measurements on the Internet. They note that for high loss rates ($p > 0.01$) that retransmission timeouts become important and the law rapidly loses accuracy. They explicitly test the case when the ratio between queuing and propagation delay is large and note that if the RTT is approximated as the propagation delay that the results are poor but that if the full RTT is included then the results are reasonable, but not as good for cases where propagation delay dominates. They attribute this to delayed acknowledgements being compressed by the long delays in arrivals. They note some concern over phase effects, although these have recently been attributed largely due to simulation environment issues rather than real-world effects [Flo02], and RED AQM is seen to produce results closer to the model. Their Internet-based experiments have more variable results with some good fits to the model and some deviating strongly. Finally the paper considers multiple congested gateways and show how by considering the delays on the inter-gateway links and the access links the throughput on each connection can be calculated with the $1/\sqrt{p}$ law and the fraction of the capacity allocated to each connection is then calculated and this agrees with the result in [Flo91]. For small access link delays and queuing delays they also obtain agreement with the TCP case (2.11 above) described in [Flo91]. This agreement of results suggests that we can apply the $1/\sqrt{p}$ law to derive network properties. In appendix B of the paper the authors note that a very slow convergence to equilibrium is often observed for TCP connections, partially due to the slow-start phase of the connections.

The previous results for the $1/\sqrt{p}$ law were rigorously proved by Padhye *et al.* [Pad00]. The method used is based on Markov renewal reward processes and allows packet losses to be correlated within one RTT but assumes that they are independent between successive RTTs. In addition to this more realistic treatment of packet losses, the method allows the authors to explicitly model the effects of delayed acknowledgement as the reduction in the average window size by a factor of $\sqrt{b}$ where b is the number of packets acknowledged by one ACK. Typcally this would be near 2. Thus they arrive at a first result, which is similar to the results above, but in terms of bandwidth rather than average window size. Then this result is extended to deal with the case of retransmission timeouts, noting that these are common in real data. Once again this result is in terms of bandwidth but we present it here in terms of average window size for ease of comparison with previous results (where T_0 is the retransmission

timeout time):

$$\overline{w_{TO}}(p) \cong \frac{1}{\sqrt{\frac{2bp}{3}} + T_0 \min\left(1, 3\sqrt{\frac{3bp}{8}}\right) p(1+32p^2)} \qquad (2.12)$$

We can see that the $1/\sqrt{p}$ still appears though there is now a second term based on timeouts. The authors then present a final result which takes into account the receiver's receive window. This can be approximated as taking the minimum of the receive window and the average congestion window at the transmitter described by (2.12). This work is then verified on a wide range of Internet experiments and is shown to be more robust than earlier versions of the law. However there is still a large variance in the results and the basic $1/\sqrt{p}$ law holds better in some cases. Padhye *et al*. refine the mathematical statement of their work in [Pad99]. They define the three terms:

- Send rate: the rate at which a source sends data.
- Throughput: the rate at which the receiver gets data.
- Goodput: the rate at which the receiver gets unduplicated data.

In all cases send rate > throughput > goodput because of packet losses and unnecessary retransmissions. They note that their previous work predicted send rates and the new more precise calculations agree with the old. The new formulae are much more complex than the old but still retain the basic $1/\sqrt{p}$ law term.

In summary we see that the TCP congestion control mechanisms of Congestion Avoidance and Fast Recovery/Fast Retransmission result in the $1/\sqrt{p}$ law with different constants depending on the version of flow control and assumptions. Modelling more TCP features such as timeouts and maximum window sizes results in additional terms but the $1/\sqrt{p}$ relationship is still present. This result has been validated over a wide range of drop probabilities and round trip times and has become the model of choice for applications such as modelling network performance (see below), multicast congestion control [WM01] and TCP-friendly congestion control [FHPW00].

This fundamental work has been extended by Cardwell *et al.* [Car00] to deal with connections that begin in slow start from a minimum window size w_1. They derive the expected duration for transferring d bytes by decomposing the transfer time into the expected time intervals spent in slow start, first loss, congestion avoidance and delayed ack phases. Here the congestion avoidance time is based on [Pad99] and simple expressions are developed for the mean time spent in the other phases assuming independent, identically distributed losses. There are a number of comparisons made with simulations and the application of just the work of Padhye et al. (i.e. assuming that the phases other than congestion avoidance have negligible impact). For small loss rates the total transfer time is shown to be dominated

by the bulk transfer in equilibrium and so there is not much difference from the earlier work. However when there is only small amounts of data to be sent the new models show the dominance of slow-start and how much poorer performance is attained than would be predicted by the earlier work. Finally there is some investigation of the distributions of the average rates and window sizes predicted but this is entirely numerical in nature.

2.4.4. The Application of Steady State TCP Throughput Models to Networks of Flows

The formal analysis of networks of TCP flows begins with Floyd [Flo91] as discussed above. The development of the $1/\sqrt{p}$ family of models for TCP steady state sending rates has stimulated a number of authors to apply these models to networks of TCP (and other) flows for modelling network performance with consequent applications in traffic engineering and network provisioning.

Key [Key98] establishes the first stage in generalising for networks of flows by treating the case for a set of *n* TCP flows sharing a single congested link. Key's major contribution is to characterise the loss rate, *p*, in terms of the window sizes, *w*, of the TCP connections sharing the congested link of capacity *C*. Previously this was a required network parameter to enable solutions. This approach was problematic as it is difficult to accurately measure the loss rate in a network [MSMO97], it is less likely that you will be able to accurately predict the loss rate for a network during design and the loss rate is dependent on the network load. Key shows that as under zero buffer assumptions the loss rate is the proportional excess of the send rate over the capacity we can solve for $\overline{w}$, the average window size of a set of TCP sources over a congested link as follows:

$$\overline{w} = \frac{1}{2}\left[\frac{C}{n} + \sqrt{\left(C/n\right)^2 + 8}\right] \qquad (2.13)$$

Simulation results are presented that show a good fit for this relationship for n flows sharing a congested link. The rest of Key's work deals with congestion pricing and so isn't discussed here.

Independently both Firoiu et al. [Fir00a] and Misra et al. [MBO99] presented a similar model for *n* TCP flows sharing a single congested link and then enhanced it by explicitly defining the average queue wait time and the relationship ("queue law") between the average queue size and the drop probability that is a decreasing function of p independent of the packet drop (queue management) strategy. The power of this approach is that a given queue management strategy, e.g. RED, will also define a relationship ("control function") between the queue length and the packet drop probability. They demonstrate that there will be a unique solution to both the queue law and the control function, which corresponds to the steady-state operating point of the system of *n* TCP flows coupled with the active queue management. They show that their model predictions give good agreement with simulation experiments. Hence,

given a queue management strategy for the congested gateway/router we can determine the average sending rate for each TCP flow, the goodput seen at each TCP receiver and the latency (transmission and queuing delays).

The final stage in the development of solutions for arbitrary topologies of flows was independently proposed by Roughan *et al.* [Rou01], Firoiu *et al*. [Fir01] and Bu *et al*. [BT01]. The new assumption presented in this work is that losses are independent of each other so that route losses can easily be calculated from link losses. This is similar to earlier work on Erlang fixed point methods [Kel86], however, instead of calculating Erlang blocking type probabilities the results for a vector analogue of (2.13) are calculated. They model a network of K links and a set of F TCP flows. Each flow traverses a path (an ordered set of links in K) within the network. For each link they specify the propagation delay and bandwidth (and in the case of Firoiu *et al*. and Bu *et al*. the queuing control function). The model then has unknowns of the link drop probabilities (and in the case of Firoiu *et al*. and Bu *et al*. the queuing delays) and the average sending rate for each TCP flow. This leads to a set of equations to be solved to find the unknowns. Flow throughputs at each link are based in the inputs from earlier links in the path and the end-to-end values for drop probability are compositions of the per link values.

This system is then solved numerically using various fixed point methods such as Newton-Raphson. The details of the treatments differ slightly with Roughan *et al.* giving the most basic approach based on early versions of the $1/\sqrt{p}$ family of models but noting that it can easily be extended to later versions and no treatment of non-TCP traffic. Modelling of queuing delays via a network of M/D/1/K queues is explored and compared with simulation results. Firoiu *et al*. add queuing delays, non-TCP traffic and issues relating to Differentiated Services and Bu *et al*. provide the most comprehensive model of non-TCP (UDP) flows. In all cases the models were confirmed by simulation experiments. The most significant problem with the approach is the lack of formally established existence or uniqueness of solutions to the system of equations solved. However all authors report rapid convergence in all networks examined (including degenerate topologies in Firoiu *et al*.). Roughan *et al.* report that the number of iterations required for convergence was both small and almost independent of network size and give advice for the choice of initial conditions. This work has similarities to the work of Gibbons *et al*. [Gib00] which was based on M/M/1/K queuing models but which lacked experimental verification.

2.4.5. Other Analytic TCP Models

The models discussed so far provide steady-state averages of sending rate, queue size and packet drop probability. They do not provide any information about the variation of these properties with respect to time or the process of the system's convergence to steady state. The transient behaviour of telecommunications networks has been shown to be very important when analysing their performance

[Fir00a, SLJ01].

Recent work on the dynamic behaviour of TCP traffic has proposed fluid models to study TCP flow properties, their stability and higher moments. An early work in this area is by Ott *et al.* [OKM96] which uses a stochastic differential equation (SDE) to model the evolution of the TCP window size over time where loss indications are described by a time-varying Poisson process. Their work captures the additive increase, multiplicative decrease behaviour of TCP but not timeouts. By rescaling the process the authors can analyse it to obtain the stationary distribution and higher moments of the average window size as a function of time. A similar approach is proposed by Misra et al. [MGT99] and extended by Altman et al. [AAB00] and Misra et al [MGT00] to include timeout events and the interaction with AQM such as RED. A disadvantage of their approach is that the intensity of loss events is no longer directly modelled but instead taken from experimental traces. This results in a set of SDEs that can be solved numerically to provide the TCP sending rate evolution in time for all flows in the network.

A major result of this work was the discovery of instability caused by RED controls. However subsequent research has shown that under more realistic network conditions these oscillations are unlikely to occur [Flo02] and that a reliance on experimental traces for modelling input can be dangerous [Pax01]. Thus these fluid models of TCP must be considered to be at an early stage of development.

There has also been work on applying traditional queuing models to TCP/IP traffic but this has largely been unsuccessful due to the closed-loop nature of TCP controls and observed properties of Internet traffic such as long range dependence (LRD) and high variability over long time scales [Arv99, Pax01]. Studies such as that performed by Roughan *et al.* [Rou01], R. Gibbens *et al.* [Gib00] and Key [Key98] have attempted to estimate queuing delays using standard models such M/D/1/K queues but their results are either not supported out by simulation studies or show poor results. A novel approach using a closed system of M/G/1 processor sharing queues by Arvidsson *et al.* [Arv01] gives a new explanation of the origin of LRD and stresses the importance of modelling the closed nature of TCP flows when estimating queue lengths. It will take time to see if this work becomes widely accepted.

2.4.6. SCTP Performance Research

As a relatively new protocol SCTP has not yet amassed a large body of research. One of the reasons for this is its heritage - industrial rather than academic personnel primarily developed it. Thus it built on practical experience with proprietary IP-based signalling products rather than academic research. During it's development the Internet Architecture Board (IAB) also insisted that as a potential bandwidth competitor with TCP on the Internet it must use TCP style rate controls. This reduced the

need for studies to demonstrate its safety to the wider Internet research community. However since it's first release as an Internet draft it has inspired a number of studies and these are briefly described here.

The first paper which addresses SCTP performance is by Jungmaier *et al.* [JST00]. This paper describes an implementation of SCTP and a number of experiments performed on a test-bed network consisting of two LANs connected by a WAN emulator. Their first experiment compares the throughput achieved over a bandwidth-delay product-limited link for a single TCP flow with SCTP flows of different user-datagram sizes. Recall that unlike the stream interface of TCP, SCTP expects all user data to be framed. They show the effects of constant message overheads in SCTP can result, for very small user datagrams in the absence of bundling, in throughputs only approximately 80% that of a TCP flow. They then demonstrate the aggregate nature of the congestion control of SCTP streams within an association and show that for any number of streams, carrying user datagrams of whatever size a single SCTP association competes fairly with a single TCP connection. In their conclusion they note that the ability of SCTP to bundle inter-network element flows into a single long-lived association with shared congestion control should ease network planning for networks of SCTP flows.

Conrad *et al.* [CHCAF01] in another early SCTP paper make reference to previous work on the ability of partially ordered protocols, such as SCTP, to improve observed throughput, delay and buffer utilisation but no specific SCTP experiments are described to support this.

There is a significant body of work dealing with the behaviour of multi-homed SCTP associations during path changeover. Iyengar *et al*. [ICAHR02] initially identified the problem of congestion window overgrowth during path changeover. This occurs due to packet re-ordering as a result of the change-over and SCTP being unable to distinguish between SACKs generated from retransmissions and those generated by transmissions. This results in TCP-unfriendly congestion window growth when the changeover is to a higher quality path (in terms of bandwidth-delay product). They present a new algorithm, Rhien, which adds information to each packet to enable the different types of SACKs to be distinguished. However they recognise that their solution is sub-optimal as it requires the definition, processing and transport of new control information (SCTP chunks). The Rhien algorithm also does not solve the problem of unnecessary retransmissions during changeover. This work is extended Iyengar *et al*. in [ICAHR03] where they propose an analytic model of the delays introduced at changeover for both transmitted packets and received SACKs. This model allows them to explore the relationship between the transmission times of the various packets and the path bandwidths and delays. NS simulations under a number of simplifying assumptions, such as no cross traffic, are used to verify this model. The authors then propose two new solutions for the problem based on making SCTP congestion control changeover aware by maintaining state for each destination in the sender. This allows the sender to refrain from unnecessarily triggering Fast Retransmit during changeover and hence unnecessarily

overloading the network and producing TCP-unfair growth in the congestion window. The second of their solutions, Split Fast Retransmit, is shown to be the more effective approach and recommended to the IETF for standardisation.

The SCTP multi-homed path changeover mechanism is also examined by Caro *et al.* [CIAHS02] but this paper examines the possibility of decoupling the SCTP failure detection mechanism from the recovery process. The authors propose that instead of the current two state SCTP model for paths (active, inactive) that a three state model be used (active, potentially inactive, inactive). Thus when a path exhibits a number of failures determined by a first threshold, α, it is marked as potentially inactive and no longer used for transmissions but is instead monitored via the standard SCTP heartbeat mechanism. If the heartbeat is acknowledged then it returns to the active state and is used once again for packet transmissions however if a second loss threshold, β, is passed then the path is marked as inactive as with standard SCTP. The advantages cited of this approach are a more rapid changeover from a path with losses, reducing the idle time necessary to trigger changeover and increasing throughput. There is no supporting experimental work to demonstrate how this would operate in practice and the authors note that as each path has its own congestion window that throughput may in fact be harmed by this rapid switching from path to path (the congestion window on an unused path rapidly collapses to 1 MTU). This work is then developed by Caro *et al.* [CIAHS02a] to also provide a comprehensive analytic model of multi-homed SCTP behaviour based on [Car00] and the author's multi-state path changeover mechanism. In the current draft form this model lacks any comparison with experimental results.

Alamgir *et al.* [AAI02] describe a series of experiments comparing SCTP and TCP congestion control over a satellite link (i.e. high bandwidth, high delay, high error rate). They evaluate the differences observed in throughput and the fairness of SCTP associations competing with TCP connections. Through simulations they demonstrate that in this environment SCTP associations have on average 20% greater throughput than SACK TCP. They improve on the earlier work of by Jungmaier *et al.* [JST00] by using a more formal definition of link sharing fairness and show that despite the increased throughput achieved by SCTP, the sharing of the used bandwidth is fair. This is attributed to the low link utilisation achieved by the competing flows due to the long propagation delays. They show that the increased throughput achieved by SCTP was due to SCTP's more robust error recovery mechanism. Specifically the quicker response to duplicate acknowledgements, the calculation of flightsize in bytes rather than segments (i.e. TCP over-estimates flightsize) and a more rapid increase of cwnd in slow-start after a loss are all cited as the cause of SCTP's improved throughput.

In a related work, Atiquzzaman *et al.*[AI03] explore the performance effects of SCTP multi-streaming in a satellite environment. They show through simulation that SCTP streams reduce head of line

blocking, goodput and the required buffer sizes at the receiver (sink). This is identified as a key requirement for small portable wireless devices such as PDAs.

Fu *et al.* [FAI02] describe the effects of delay spikes in a wireless mobile environment on SCTP, Reno TCP and Eiffel TCP. They show through simulation that SCTP and Reno TCP are vulnerable to spurious timeout and unnecessary retransmission events being generated by delay spikes. In the absence of packet losses during the delay spike both SCTP and Reno TCP perform similarly but are both worse than Eiffel TCP. When packet losses are introduced then the Eiffel algorithm fails and both SCTP and Reno TCP achieve higher throughputs. The operation of SCTP SACKs is shown to be superior to Reno TCP for scenarios involving large receiver windows and bandwidths but Reno TCP performs better for small receiver windows. The authors then recommend using different transport protocols based on the anticipated properties of the operating environment.

Fu *et al.* [FA03] investigate the effect of SCTP's enhanced ability to report non-consecutive packet losses in SACKs when compared to SACK TCP and Reno TCP in a Mobile IP environment. They characterise a handover in a Mobile IP environment as one in which multiple non-consecutive packet losses within a single window are likely. Through simulation of a number of network configurations they compare the performance of SCTP and SACK TCP in a Mobile IP environment during handover events. The ability of SCTP to report a large number of non-consecutive packet losses through Gap ACK reports in the SACK chunk results in SCTP gaining higher throughput than SACK TCP or Reno TCP. This performance gain is most pronounced at lower wireless bandwidths where fast retransmission events dominate rather than retransmission timeouts. Ye *et al.* [YSL02] have analysed the interaction between SCTP congestion control and IEEE 802.111 wireless LAN protocol used as a link layer. As shown in earlier work on TCP [XS02] the high-loss wireless link layer causes the transport layer to throttle back on throughput as it misinterprets the losses as network congestion events.

Ladha et al. [LACI03] demonstrate via simulation that FTP over SCTP achieves higher throughputs than Reno TCP, especially in an environment where there is packet loss.

2.5. Summary and Conclusions

This chapter began with a survey of the evolving telecommunications services domain and identified both driving forces for change and current key technical challenges for network operators and equipment vendors. Then a brief survey of the CORBA middleware platform and SCTP signalling transport protocol were presented with particular emphasis on the CORBA networking architecture and SCTP congestion control mechanisms. These descriptions highlight the desirability of integrating such technology in the telecommunications services domain but also the relative immaturity of the

technology when applied to real-time telecommunications applications.

The world of IN standardization and evolution is changing. There are many disparate bodies involved in the field, a mix of both traditional telecommunications and information technology solutions will soon be widely deployed. There will be both successful and unsuccessful specifications but there are several underlying trends in the IN evolution path that will be present in future systems. Due to the energy being put into new technologies and their ability to open new markets it is likely that the pace of change will continue to accelerate.

CORBA (or some form of middleware) is seen as necessary glue to bind these new architectures together with a manageable programming model. It also opens up the possibility of combining the two traditionally separate management and control planes of telecommunications software. Current solutions for middleware-based approaches suffer from the following common failings:

- They emphasise innovation over integration so leave unanswered important real-world questions of interoperability.
- CORBA platforms themselves are relatively unproved in the telecommunications domain.

A key requirement for signalling systems is reliable message transport that is supported by an analytic understanding of the protocol to support network planning and traffic management tasks. This is also a necessity for any transport to be used in conjunction with a middleware platform. The new SCTP transport protocol is designed to fulfil this role but has the following limitations:

- It has not been widely deployed or tested and so requires rigorous investigation for protocol errors before it can be accepted by the telecommunications community.
- There is no analytic framework for performing network planning of multi-homed hosts.

Chapter 4 outlines a comprehensive architecture for creating CORBA-based systems of IN entities, this is then generalised to allow any TC-User protocol to be mapped to entities in the CORBA domain. This enables heterogeneous multi-protocol, combined service and management applications to be built in the CORBA domain.

Chapter 5 investigates SCTP as a potential protocol for CORBA telecommunications service objects to communicate with. Flaws in the original SCTP specification are discovered through experiments and the properties of the protocol, especially in multi-homed environments, are explored through the development of a single association analytic model.

Chapter 6 expands this work by providing an analytic modelling framework for arbitrary networks of multi-homed SCTP signalling nodes. This is then combined with a model of the CORBA-based Distributed Intelligent Network to evaluate the likely performance of such a system.

Chapter 3. Methods and Tools for Architectural Development and Performance Analysis

In this chapter we provide a review of methods and tools used for the main body of our work. This is naturally split into two parts; §3.1 dealing with object-oriented distributed systems and §3.2 dealing with network modelling techniques.

3.1. Object-Oriented Distributed Systems

An OO distributed systems approach to telecommunications service networks is at the core of this thesis. In this section a brief overview of OO terminology, notation and methodology is presented.

3.1.1. Object-Oriented Systems

Software system design suffers from an inherent complexity that derives from four elements, the complexity of the problem domain, the difficulty of managing the development process, the flexibility of software systems and the problems of characterising the behaviour of discrete systems. The relative maturity of the telecommunications domain has multiplied the already complex mix of competing requirements with an ever-increasing diversity of "legacy" systems to be incorporated, standards to be adhered to and new technology to be incorporated.

Researchers such as Booch [Booch91] noted that complex systems occur in many fields and have common properties such as being composed of a hierarchy of interrelated subsystems. The choice of what constitutes a fundamental component of a system is largely arbitrary and dependent on the perspective of the observer. The level of interdependency between subsystems is much less than the interdependency of components within a subsystem. In addition, they noted that two types of hierarchies of components were of particular value when modelling. A *structural* hierarchy that defined what components were building blocks for other components (an *object* structure) and a taxonomy that identified groups of components with common behaviour (a *class* structure). Their solution to the problems of software complexity was to develop Object Oriented Analysis and Design methods that decompose and characterise systems in these ways.

Classes are used to specify object types within the class structure. Relationships can be established between classes that collaborate to perform some higher-level component or sub-system. One special

type of relationship is a *generalisation* whereby a relatively abstract base class is extended by making it tailored to a more specific need. Each object in a software system has two key aspects, an *interface*, which defines the services that the object can offer, and an implementation which is the actual code that does the work behind the interface.

Object-oriented technology is one of the key drivers behind distributed systems because objects are natural elements of distribution across a network. Objects fulfil this role well because they are naturally partitioned elements of code plus data and they have well defined application level interfaces through which they communicate. Prior to the advent of distributed object technology, distributed systems were built by adding specialised interfaces onto the parts of a system that needed to communicate. Object-based systems are inherently defined in terms of interfaces.

3.1.2. Specification Languages and Methodologies

In the course of the work described in this thesis several OO notations and methods were used. This section gives a brief overview of these topics.

3.1.2.1. CORBA IDL

CORBA IDL is the CORBA metadata language for self-description of any CORBA system. The self-descriptive nature of CORBA systems enables independently developed components to dynamically discover how to collaborate. This is of special importance in a highly heterogeneous environment like a telecommunications network. The Interface Repository is the CORBA metadata repository. It is normally implemented as a runtime database that contains the interface specifications of each object an ORB recognises.

IDL is a contractual language for the specification of a CORBA server object's boundaries and interfaces with potential client objects. It is programming language neutral and declarative in nature i.e. it can't be used to specify implementation algorithms, just interfaces. All objects on the CORBA bus have their service interfaces specified in IDL. These service interfaces include a description of any resource or service that a server object exposes to its clients. IDL can specify attributes, parent classes, exceptions raised by the object, typed events, globally unique interface and method identifiers and the methods supported by an interface – including the method's typed parameters. The grammar used is a subset of C++ grammar with additional keywords to support distributed processing concepts. Unlike many specification languages IDL has a set of standard programming language mappings that directly link the software architecture, defined by its interfaces, and the compiled code that implements it.

The structure of CORBA IDL

Figure 3.1 shows the main elements that constitute an IDL specification. At the top of the figure we see the *module* keyword that is used to define a namespace for a group of class (interface) definitions.

Module names are scoped by a series of one or more identifiers separated by the characters "::". Then a series of module-wide declarations may be made. The next feature is an *interface* definition which groups a set of *operations* (methods) that a client can invoke on a CORBA server object which supports this interface (class). Within the interface one can declare interface-wide constructed data types, constants, exceptions and attributes. Attributes are types for which get and (optionally) set methods will automatically be generated. Interfaces may inherit from one or more parent interfaces. Within each interface a series of operations are normally defined to denote the services that the interface supports. Each interface is generally mapped to a class definition by a language binding which the implementation then inherits from in order to realise the interface. This generated code is known as either a skeleton code for a server object or stub code for a client object.

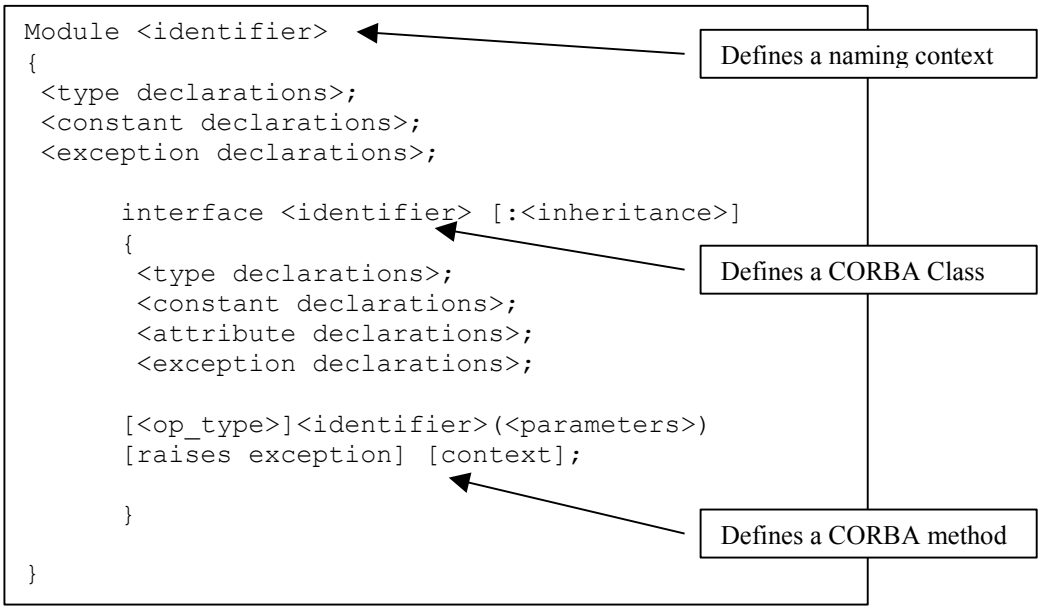


Figure 3.1: Main elements of a CORBA IDL file

Each operation can be defined in terms of its complete signature including typed parameters and a return value. One or more exceptions that can occur during the processing of an operation can optionally be defined. These user-defined exceptions are always in addition to standard CORBA exceptions dealing with various failures of the distributed computing environment. Operation *context* attributes may also be defined to describe a CORBA client's local environment but this feature is used infrequently. Each operation maps to a method definition in either skeleton or stub code via a specific language binding.

Throughout the module definition CORBA *types* are used to describe data. These may be basic types such as short or long integers, constructed types such as sequences and unions or object references to other CORBA objects. CORBA includes a special data type called an "any" which is useful in dynamic

situations as it allows you to represent any possible IDL type. Each CORBA IDL data type is mapped to a native data type via the appropriate language bindings. Every data type defined in IDL also has a unique *type-code* generated for it. This type-code is used in inter-ORB protocols, dynamic invocation interfaces and the any data type to create self-describing data which can be passed across operating systems and ORBs.

3.1.2.2. UML

The Unified Modelling Language (UML) is the successor to the group of object-oriented analysis and design (OOA&D) methods that were pioneered in the late 1980's and early 1990's. It is most directly based on the work of Booch, Rumbaugh and Jacobson but is now an OMG standard with many contributors. OOA&D methods normally encompass both a modelling language (set of notations) used to express designs and a process of steps to take during a design. Although derived from work on methods, the UML is a language rather than a method. In many ways having a common modelling language simplifies the OOA&D world by giving us a common means of expressing and communicating ideas or designs. Processes may then be developed independently and tailored to specific needs or goals.

The UML specifications define both a notation and a meta-model. The meta-model can be used to formally define the meaning of the notation and to allow checking that a particular design using the notation is syntactically correct. UML is highly modular and adaptable with features that allow new notation to be imported into the language. In addition there are generally multiple ways to represent a given design feature. In part, this is due to the heritage of UML's predecessors as separate methods and partially because having multiple ways to view a system can be useful for analysis and design. These properties make it hard to identify the core of UML but traditionally the following major features are used in most software development projects:

- Class Diagrams: To identify the types of abstractions used in a system and to identify their relationships.
- Interaction Diagrams: To illustrate the key behaviours of a system.
- Use Case Diagrams: To ease communication with domain specialists when gathering requirements for the system.

There are also a number of less often-used UML features such as package diagrams, activity diagrams, deployment diagrams and state diagrams. Each of the major features of UML are briefly discussed in the following sections.

Recently UML has attained popularity as a specification language for the development of standards such as at the 3GPP and Parlay consortia. Apart from the obvious advantages of using a well-

understood specification language, the reason for this is that specifications can be made in UML in a technology and implementation independent way. Despite this technology independence there are also well-defined mappings of UML specifications to CORBA, DCOM or other distributed systems technologies. Thus common specifications can be developed which equipment vendors can implement according to their own desired distributed systems technology or programming languages.

Use Cases Diagrams

Use cases are formed from sets of *scenarios* that describe a sequence of interaction steps between a user and a system. A set of scenarios in which the user has a common goal forms a use case. They are extensively used in the requirements capture phase of a project. A simple format for use cases is a textual description with numbered steps. UML specifies notation for use case diagrams, which help visualisation and allow the description of various types of relationships between use cases.

Class Diagrams

Class diagrams were at the core of all the methodologies that preceded UML. In essence they describe the types of objects in the system and the various kinds of relationships between them. Class diagrams are a very versatile tool that allows modellers to express system concepts from several different perspectives. They can be:

- Conceptual: Diagrams that represent the concepts in the domain under study. These concepts relate to the classes that implement them but often there is no direct mapping. This form of class diagram is often used during requirements gathering an analysis.
- Specification: A model of a software system that emphasises interfaces rather than implementation. This is used during architectural design or specification. It is the type of class diagram that most naturally maps onto CORBA IDL specifications.
- Implementation: An implementation model of a system. These detailed models can be used to generate source code in OO CASE tools.

In general class diagrams should be created and interpreted within one perspective.

The basic notation of class diagrams is illustrated in figure 3.2. It shows a fragment of the specification of the base interfaces for the CORBA-TC mapping. The *association* between the TcUserGenericFactory and TcUser represents a relationship between instances of these classes. Each end of the association also has a *multiplicity*, which is an indication of how many objects may participate in the given relationship. The "*" on the TcUser end of the association indicates that a TcUserGenericFactory may have many TcUsers associated with it, whereas the "1" on the other end indicates that TcUser is created by only one factory. Associations may be named or have an arrow to indicate the direction in which responsibility flows.

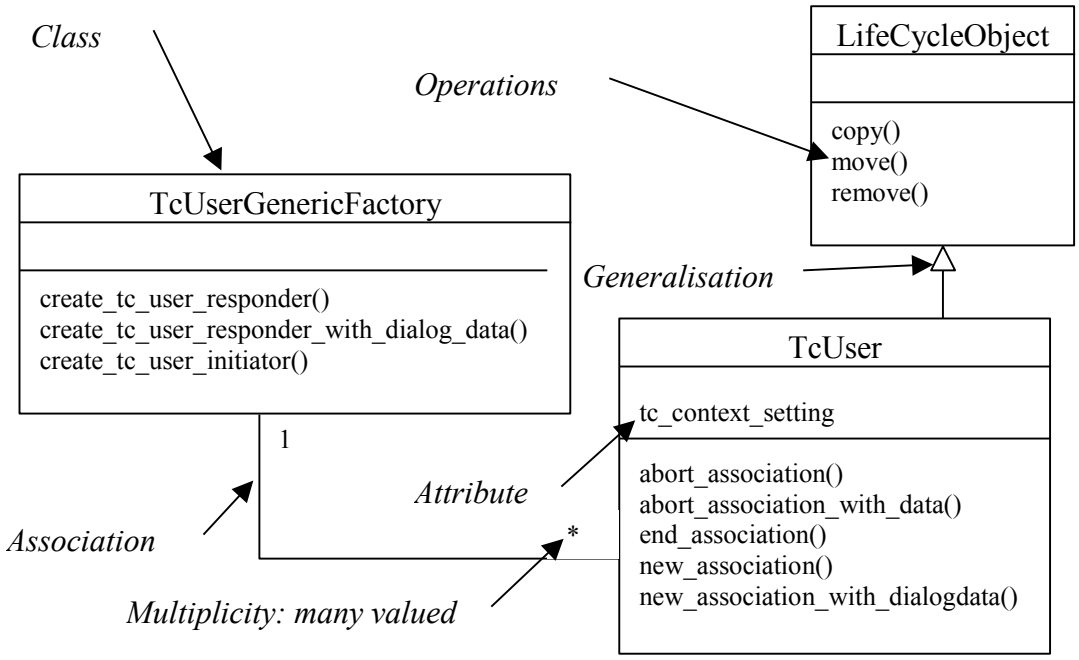


Figure 3.2: UML class diagram notation

Each panel representing a class is divided into three sections, which are from top to bottom, the class *name*, *attributes* and *operations*. The name is self-explanatory and provides an identifier for the class. Attributes indicate a property value of a class that can be queried. More explicit notation can be used to specify the type, *visibility* and default value for an attribute. Briefly the visibility of an attribute or operation (see below) defines the range of objects or classes that can access that attribute. Operations define the services that an object can provide. Full operation signatures may be specified in the UML that define the visibility, parameter list (optionally including any defaults) and return type of an operation.

The final UML notation illustrated in figure 3.2 is *generalisation*. Within a specification model, the perspective used in this document, generalisation means that the interface of the subtype must include all elements from the interface of the supertype. This is both a convenience in notation to avoid repeating common specification elements and a modelling tool showing families of related entities that can be manipulated in similar ways.

Interaction Diagrams

Interaction diagrams are models that describe how groups of objects collaborate in some behaviour. Typically they are used to capture the behaviour of a single use case. There are two types of interaction diagrams: sequence diagrams and collaboration diagrams. Sequence diagrams are closely related to traditional telecommunications specification and design language (SDL) [SDL92] message sequence charts (MSCs). Hence there is little need to illustrate the minor differences in notation between them.

Collaboration diagrams are related to class diagrams with numbered message passing for the current use case indicated on the class structure diagram. Figure 3.3 shows an example of a collaboration diagram.

The most significant new notation shown here is the UML object-naming scheme which takes the form *objectName* : *ClassName,* where either the object name or the class name may be omitted. This naming scheme is also applicable in class diagrams and sequence diagrams.

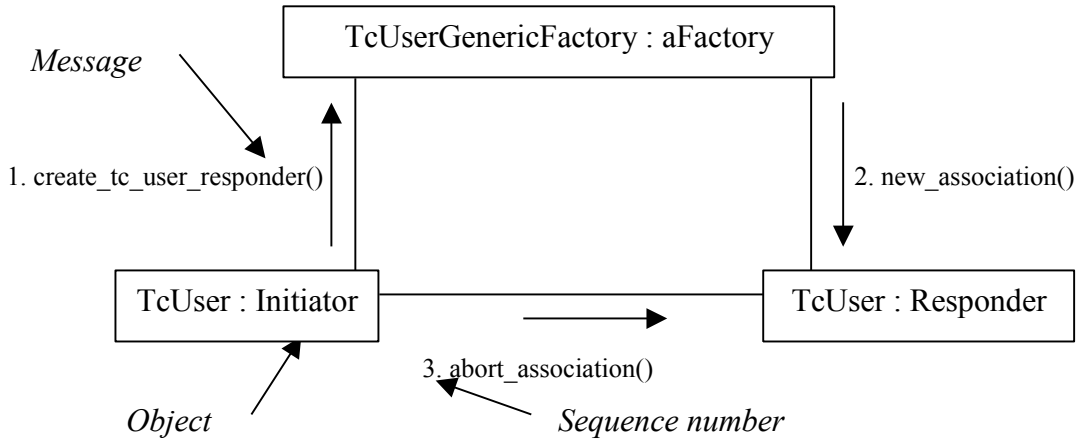


Figure 3.3: UML collaboration diagram notation

3.1.2.3. Design Patterns

The OO design pattern movement traces its origins to Christopher Alexander, an architect, who first applied the idea of design patterns to the physical architecture of buildings and urban development. Design patterns [GHJV94] describe simple and elegant solutions to specific problems in object-oriented design. The huge variety of options available for structuring OO systems often lead to sub-optimal design by inexperienced designers. Design issues that must typically be addressed are: finding objects, factoring them into classes at a useful granularity, defining class interfaces and inheritance hierarchies and establishing key relationships between these entities. Experienced designers have discovered many recurring types of problems in OO design that both solve the immediate requirements but also maximise elegance, flexibility and reusability. These solutions have been refined through much iteration and often make sophisticated use of OO features in unexpected ways.

Design patterns essentially document the experience and techniques of earlier developers and make them more accessible to developers of new systems. Thus, when confronted with a particular design problem there is a catalogue of potentially applicable techniques to guide a developer's design. In addition, by naming each pattern the vocabulary of OO design is enriched, allowing design to take place at a higher level of abstraction. In general a design pattern has the following elements:

- **A pattern name:** This acts as a simple description of the collection of attributes of a design problem, its solutions and their consequences.
- **The problem statement:** This describes when to apply the pattern by explaining the design problem and its context. It may give guidelines for when the pattern should not be applied.
- **The solution:** This part of the description lists the elements that make up the pattern, their relationships, responsibilities and collaborations. It is important to note that because the pattern is a template the solution doesn't describe a concrete implementation but rather an abstract description showing the general arrangement of elements and how they are used to solve a general problem.
- **The consequences:** This section discusses the results and trade-offs of applying the pattern. This can be critical for evaluating design alternatives. Software trade-offs often concern storage space and execution time but design concerns such as impact on system flexibility, extensibility and portability may also be discussed.

Patterns may be applied at various stages of the software development process, thus we can speak of Analysis Patterns, Architectural Patterns, Design Patterns, etc. Design patterns are descriptions of communicating objects and classes that are customised to solve a general design problem in a particular context.

An example of a design pattern used in chapter 4 is the "Factory Method" class creational design pattern. This provides a common interface for creating an object but defers the decision on which class to instantiate to a subclass. This allows the construction of generic gateways framework that knows nothing of the specific TC-User protocols supported.

3.2. Network Modelling

As systems, like modern computer networks, become more complex it is increasingly difficult to understand and predict their behaviour. These predictions are necessary to evaluate new technology, to manage the evolution of deployed systems and to estimate or control the performance these systems. Traditionally intuition built up over years of experience with a system is a strong guide to its behaviour but this is expensive to acquire and often unavailable. Experimentation in controlled environments can also be very useful but is generally time-consuming and expensive. Finally it may not be possible to create realistic experiments in some circumstances due to the shear scale of the system being studied.

Modelling is a third approach that utilises a defined abstraction of the system being studied to predict the system's behaviour. Once defined, the model can be parameterised to reflect alternative scenarios under study and then evaluated to determine the behaviour for a given alternative. Modelling compliments the activities of intuition and experiment as it can be used to quickly explore scenarios suggested by intuition or to structure a series of experiments to ensure the most important alternatives are explored.

There are two main approaches to modelling activities, firstly analytic modelling whereby a mathematical formulation of the abstraction (model) is created which can then be solved using either deductive reasoning or numerical analysis for cases where exact solutions are unobtainable. Secondly event-based simulation models can be used whereby more detailed models are constructed as computer programs and the evolution of the system in time is computed. In the following sections we discuss each in turn.

3.2.1. Analytic Network Modelling

The most common field of mathematics applied to communications network modelling is queuing theory, an application of the theory of stochastic processes to the analysis of queuing systems. This approach has served the telecommunications sector well for many years. However the violation of many queuing theory assumptions by data communications protocols such as TCP has led to a resurgence in analytic network modelling approaches which are built on basic probability theory and the theory of stochastic processes rather than the full range of tools of queuing theory.

3.2.1.1. Basic Probability Theory

Classically the probability of an event A occurring, P(A), is based on counting the number of possible outcomes of an experiment in which the event occurs and dividing this number by the total number of equally likely possible outcomes of the experiment. However, this definition gives no guidance on the appropriate selection of the set of equally possible outcomes. If we instead consider an experiment that can be repeated a number of times where each performance of the experiment is called a *trial*. Then we can define the *statistical probability* of A occurring in a given trial as

$$P(A) = \lim_{M \to \infty} \frac{m}{M} \qquad (3.1)$$

Where m is the number of occurrences of A in M trials. This more flexible, though still imperfect, definition of probability is the one that we shall use.

In general the outcome of a random experiment can be denoted by ω, an element of a set of possible outcomes, Ω, called the sample space. Events are subsets of Ω and we will generally want to assign probabilities to collections of events. A probability measure, P, is a function that assigns a *probability* (a non-negative number) to each event in a collection of events. We often find the probability of a given event by expressing it in terms of *conditional probabilities*, i.e. the probability of an event given that another event has occurred. Bayes' Theorem (3.2) gives us a way to relate conditional events: For event B and sequence of events $\{A_i\}$,

$$P(A_i \mid B) = \frac{P(B \mid A_i)P(A_i)}{P(B)} \tag{3.2}$$

We define events to be *independent* if the probability of A occurring is unchanged by the knowledge that B has occurred. This is a symmetric pairwise relation between the events. This property is strengthened to the events being *mutually independent* if every finite subclass of a class of events are also independent.

A *random variable* is used to represent a measured quantity in a trial or experiment. It is a real-valued point function of ω, the possible outcomes of the trial. This differs from a probability measure which is a set function which characterises events which are themselves subsets of the sample space Ω. We can define probabilities to events based on the value of a random variable. An important example of a random variable is an *indicator function* which takes a value of 1 if the event being tested for (indicated) occurs and a value of 0 otherwise. If we have a random variable *X*, then we can define a cumulative *distribution function*, *F*, of *X* where

$$F(x) = P(\{\omega: X(\omega) \leq x\}) \tag{3.3}$$

Where the explicit mention of ω is usually omitted. It is often convenient to work with the *tail distribution* (or *tail*) of *X*, which is $P(X > x) = 1 - F(x)$. Distribution functions are often used to define random variables.

Indicator functions as a type of random variable have already been mentioned. They belong to a class of *discrete* random variables in which the value of the variable has a finite set of values. Each value of the random variable has a defined probability between which *F(x)* is flat. The *binomial, geometric* and *Poisson* distributions are all examples of discrete distributions. The form of the Poisson distribution is given here due to its importance in later sections:

$$P(X = x) = \frac{\lambda^x e^{-\lambda}}{x!}, \quad x = 0, 1, \ldots \tag{3.4}$$

Where λ is both the mean and variance of the distribution.

Alternatively random variables can be *continuous* if *X* has no values *x* with the property *P(X = x)* > 0. In this case it is usually assumed that a probability density function (PDF), *f(x)*, exists which allows us to describe probability as the area under a curve. Examples of continuous distributions are the *normal*, *uniform* and *exponential* distributions. The exponential distribution is defined as follows:

$$\begin{aligned} f(x) &= F'(x) = \mu e^{-\mu x}, \quad x \geq 0 \\ f(x) &= 0, \qquad\qquad\quad x < 0 \end{aligned} \tag{3.5}$$

Where the mean of the distribution is $1/\mu$ and the variance is $1/\mu^2$.

This distribution has the important additional property that it is *memoryless*. This means that the occurrence of any value of *x* doesn't change the distribution of expected outcomes.

For discrete random variables we can define the *expected value* (or mean) of *X*

$$E(X) = \sum_x xP(X = x) \tag{3.6}$$

This is a measure of the location or magnitude of the distribution. This is simply the average of the possible values of *X* weighted by their probability of occurring. It is also known as the *first moment* of *X*. If *X* is continuous with density *f(x)*, the expected value of *X* is

$$E(X) = \int_{-\infty}^{\infty} xf(x)\,dx \tag{3.7}$$

If *X* is a nonnegative random variable then *E(X)* can also be calculated from the tail distribution of *F(x)*. The expectation values of random variables and constants are easy to combine with each other and to calculate it from its own distribution.

We define the *variance*, a measure of the variable's distribution around the mean in terms of expectation values as follows:

$$\sigma^2(X) = E\{[X - E(X)]^2\} = E(X^2) - [E(X)]^2 \tag{3.8}$$

The standard deviation of *X* is defined as the square root of the variance.

If we can assign probabilities to events of the form $P\{u(X) \leq u\}$ then functions *u(X)* of random variables are themselves random variables. An important class of functions of random variables are their *generating functions*

$$G(z) = E\{z^X\} = \sum_{i=0}^{\infty} p_i z^i \tag{3.9}$$

These allow us to easily calculate the factorial moments of the random variables. Each distribution has its own characteristic generating function. If we have a random variable, *U*, which is the sum of independent random variables we can conveniently calculate the generating function of *U,* from the product of the generating functions of the constituent random variables. An alternative method of finding the higher moments of a distribution is to use the Laplace transform of the distribution's density function.

When dealing with the value of a random variable measured in a large series of independent trials we assume that the average measured mean of the random variable is likely to be close to the expectation value of the random variable. We can use the *laws of large numbers* to make precise statements about how and when the average measured mean approaches the expectation value. The *weak law of large numbers* states that: For a sequence $X_1, X_2, \ldots$ of independent, identically distributed (i.i.d.) random variables with finite mean μ and variance σ^2,

$$\lim_{n\to\infty} P(|\overline{X}_n - \mu| > t) = 0, \quad \text{for any } t > 0 \tag{3.10}$$

Thus as the number of trials approaches infinity $\overline{X}_n$ converges in probability to μ. The *strong law of large numbers* strengthens this condition by generalising from the specific case of an *n* to making the same claim for all sufficiently large *n* simultaneously. It states that: For a sequence $X_1, X_2, \ldots$ of independent and identically distributed (i.i.d.) random variables with finite mean μ and variance σ^2,

$$P(\{\omega\} : \lim_{n\to\infty} \overline{X}_n(\omega) = \mu) = 1 \tag{3.11}$$

In fact both of these laws also hold under weaker conditions where the random variables are merely uncorrelated rather than independent and the variance of the variables may be infinite. This leads to the general result that for finite constant μ

$$\overline{X}_n \to \mu \quad \text{w.p.1} \Rightarrow E(X_n) = \mu \tag{3.12}$$

In many cases we deal with a sequence of random variables (Y_n, n = 1, 2,…) that converge to a random variable. We say it is *convergence in distribution* if, where F_n is the distribution function of Y_n and

$$\lim_{n\to\infty} F_n(t) = F(t) \tag{3.13}$$

For all points *t* where *F* is continuous. The *central limit theorem* uses convergence in distribution to relate the number of samples, mean and variance of a sequence of i.i.d. random variables to the normal distribution.

3.2.1.2. Stochastic Processes

A *stochastic process* is an infinite collection of random variables that may be used to model some real world phenomenon such as the arrival process of customers at a queue. To build a *stochastic model* we define the interacting processes and the nature of their interaction. We index the collection of random variables in a stochastic process in some way, e.g. $\{X(t): t \in T\}$ on an index set T. The index set is

defined over an interval for a *continuous time* process or over a countable set of nonnegative integers for a *discrete time* process. Each possible value of $X(t)$ corresponds to a *state* of the process. A stochastic process is well-defined if the joint distribution of $X(t_1)$, …, $X(t_k)$ is well defined for every finite set of times t_1, …, t_k. The stochastic process is also defined on some sample space Ω. For any fixed point, ω, in our sample space every random variable is a real number and every event either occurs or does not occur. Thus we could more accurately say that our stochastic process is dependent on ω as well as *t*. For fixed ω the process defines a *sample path* of the process. We can also distinguish between processes that are made up of discrete-valued random variables or continuous-valued random variables. These define *discrete-state* and *continuous-state* processes respectively. For the special case of discrete state processes where the random variables are non-negative integers we may define a *renewal process*.

If we consider events called *renewals* that occur over time with the times between successive events represented by independent non-negative random variables, then we have a renewal process. If we look at the limiting behaviour of this process (as time goes to infinity) then we see *time averages* of the process. If we have a complex process that at some time starts over or "renews" itself then we have a *regenerative* process. For convenience we usually assume that the time before the first event can have a different distribution from the inter-event time distributions. Properties of this process include

$$Z_n = \sum_{j=1}^{n} X_j, \quad n = 1, 2, ... \tag{3.14}$$

The time (epoch) of the *n*th event, where the X_j are the random variables defining the inter-event times. The *rate,* μ, at which events occur is the reciprocal of the mean (expectation) of the inter-event times. We can also examine the distribution of events in time by using the convolution of the inter event time distributions. We may wish to know the number of renewals that have occurred by a certain epoch, $t \geq 0$,

$$M(t) = \max\{n \geq 0 : Z_n \leq t\} \tag{3.15}$$

For each *t*, this is an integral-valued random variable. Over all *t* this is a continuous-time stochastic process called a *renewal* process. We can also define a *renewal function* $m(t)$ which defines the expected value of $M(t)$ in terms of the distribution of the inter-arrival times. It can be shown that independent of the distribution functions of the X_j that

$$\lim_{t\to\infty} \frac{M(t)}{t} = \mu \quad \text{with probability 1 (w.p.1)} \tag{3.16}$$

This is the Elementary Renewal Theorem. This is true for both discrete and continuous (or mixed) inter-

event distribution functions. Other properties of the process can also be described in terms of the number of renewals by a certain epoch, $M(t)$, for example the *excess*, or time until the next renewal $Y(t)$ and the *age*, or time since the last renewal, $U(t)$.

We can enhance the renewal process model by adding the concept of a reward or cost associated with each renewal. These can be defined in terms of the cumulative reward, $C(t)$, in which case we have a *cumulative* process. Where the average reward is give by the *Renewal-Reward Theorem*:

$$\lim_{t\to\infty}\frac{C(t)}{t}=\frac{E(R_j)}{E(X_j)} \quad \text{w.p.1} \qquad (3.17)$$

Where R_j are the random variables that define the rewards associated with the events occurring at times X_j.

3.2.1.3. The Poisson Process

A Poisson process is a stochastic renewal process $\{\{\Lambda(t):t\geq 0\}$ where $\Lambda(t)$ can be interpreted as the number of events in [0, t]. Each $\Lambda(t)$ has a Poisson distribution. The inter-event times of a Poisson process are i.i.d. exponential. It is important for modelling phenomena like arrivals in queues. Viewed from the point of view of a fixed ω it is a non-decreasing step function. The number of arrivals in an interval (t, $t + h$] is called an increment. As the number of arrivals in disjoint intervals can be described by independent random variables a Poisson process is said to have *independent increments*. Thus for each t, $\Lambda(t)$ has a Poisson distribution with mean λt, where t is the interval length and λ is the arrival rate. The third property of a Poisson process is that the distribution of the increment is independent of the epoch t where the increment is located. As the size of the increment diminishes so does the probability of two or more events occurring in that increment, thus we say that the process has *orderliness* i.e. events occur one at a time. Finally, the probability of one arrival in a short interval is approximately equal to the length of an interval. These are the properties of a Poisson process that make it so useful for modelling things like the arrival of telephone calls at an exchange.

Some consequences of the above properties of Poisson process are as follows. Given exactly one arrival during [0,t] then the arrival epoch is uniformly distributed on that interval. Poisson processes can themselves be randomly partitioned into independent Poisson processes or the sum of independent Poisson variables is Poisson.

Rather than focusing on the inter-event times of a Poisson process we can instead define a random variable T, the stopping time for the process, which defines the epochs at which the process renews itself. Each stopping time is dependent only on the process up to the current epoch, t, and has no dependence on the future increments of the process. If T is the stopping time of a Poisson process

$\Lambda(t)$ with rate λ then $\{\Lambda(t+T)-\Lambda(t): t>0\}$ is also a Poisson process with rate λ that is independent of T, $\{\Lambda(t): 0 \le t \le T\}$.

Poisson processes may be generalised into *point processes* by relaxing some of their properties. A renewal process allows inter-arrival times to have arbitrary distributions. A *batch* Poisson process no longer has the orderliness property and so arrivals from this process may occur in batches.

We can generalise the concept of Poisson process stopping times to *regenerative* processes. These are stochastic processes $\{B(t): t \ge 0\}$ which start over at some random epoch X_1 after which the process is independent of the past and has the same structure that it had originally, i.e. is stochastically equivalent. X_1 is called a *regenerative point*. Due to the fact that $\{B(t)\}$ regenerates at X_1, there must in fact be a series of subsequent i.i.d. regeneration points, X_j, which are called the cycle lengths. $\{X_j\}$ is itself a renewal process which is embedded in $\{B(t)\}$.

3.2.1.4. Markov Processes

The Markov property is used to define an important class of stochastic processes called *Markov processes*. This states that given the present state of a process the future evolution of the process is independent of the past evolution of the process. A Poisson process has the Markov property and is therefore a simple type of Markov process. Markov processes can be discrete or continuous with respect to their state-space or time, we deal with discrete state-space Markov processes which are also called *Markov chains*.

Formally a sequence of integral-valued random variables $\{X_n: n = 0, 1,\ldots\}$are a discrete Markov chain if they have the Markov property:

$$P(X_{n+1}=j \mid X_n, X_{n+1},\ldots,X_0) = P(X_{n+1}=j \mid X_n), \quad \begin{matrix} n=0,1,\ldots \\ j=0,1,\ldots \end{matrix} \qquad (3.18)$$

where, for each value of X_n, the probabilities are called *transition probabilities*. We will assume that the transition probabilities are stationary i.e. they are independent of the epoch, n. We can denote the 1-step transition probability matrix for the system as follows:

$$P=(p_{ij}) \quad \text{where } p_{ij}=P(X_{n+1}=j \mid X_n=i), \quad i \ge 0,\ j \ge 0 \qquad (3.19)$$

P is a square matrix of non-negative elements with row sums equal to 1. We can also build an n-step transition matrix, $P^{(n)}$, containing the probabilities that transitions will occur across multiple states. In general $P^{(n)} = P^n$, $n \ge 1$.

This is related to renewal theory because the times (numbers of transitions) between returns to a particular state are i.i.d. random variables. This is a consequence of the Markov property. We can

define the distribution of first passage time between arbitrary pairs of states, $i \geq 0, j \geq 0$, as

$$f_{ij}^{(n)} = P(X_n = j, X_{n-1} \neq j, ..., X_1 \neq j \mid X_0 = i), \quad n = 1, 2, ... \qquad (3.20)$$

Hence we can define a first passage time random variable, T_{ij}, where $P(T_{ij} = n) = f_{ij}^{(n)}$, $n = 1, 2, ...$ with $f_{ij}^{(n)}$ as its distribution. If $i = j$ then a visit is called a *return*. States which a chain can never leave are called *absorbing* and have $p_{jj} = 1$. *Recurrent* states have $f_{jj} = 1$ and transient states have $f_{jj} < 1$. For recurrent states we can define a *mean recurrence time* for state j, v_j

$$v_j = E(T_{jj}) = \sum_{n=1}^{\infty} n f_{jj}^{(n)} \qquad (3.21)$$

Which may be either finite or infinite. State j is called *positive* if $f_{jj} = 1$ and $v_j < \infty$ and *null* if $f_{jj} = 1$ and $v_j = \infty$. If $M_{ij}(n)$ is the number of visits to state j by epoch n then $\{M_{ij}(n)\}$ is a renewal process.

If there is a positive probability of a transition between two states then they are said to be *reachable*. This implies that there is a path of positive probability transitions between the two states with a *length* equal to the number of 1-step transitions required. Two states are said to *communicate* if state i can reach state j and vice versa. The set of all states in a system has subsets called *communication classes* that contain groups of communicating states. Any subset of states is said to be *closed* if none of the states in that subset communicate with states outside the subset. A Markov chain is said to be *irreducible* if all its states communicate.

Communication classes have a number of useful properties. The states in a class are all positive, all null or all transient. Open classes are always transient. Finite closed classes are always positive. We can represent Markov chains as state diagrams as shown in figure 3.4.

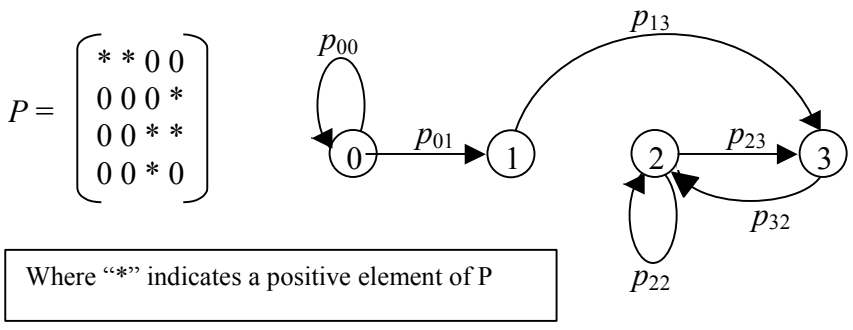


Figure 3.4: Markov chain as a state diagram

Birth-Death Processes

A birth-death process is a (discrete- or continuous-) time Markov chain in which state transitions only take place between neighbouring states. For a discrete state-space defined by the set of integers then this birth-death property requires that if $X_n = i$, then $X_{n+1} = i - 1$, i, $i + 1$or no other value. Hence, birth-death processes provide a means of modelling a system where the time intervals approach zero such that only a single event (an arrival or a departure) occurs during an interval.

The probability of a birth-death process being in a particular stake k at time t is denoted by $P_k(t)$ where:

$$\begin{aligned} P_k(t+\Delta t) &= P_k(t) - (\lambda_k + \mu_k)\Delta t P_k(t) + \lambda_{k-1}\Delta t P_{k-1}(t) + \mu_k \Delta t P_{k+1}(t) + o(t) \quad k \geq 1 \\ P_0(t+\Delta t) &= P_0(t) - \lambda_0 \Delta t P_0(t) + \mu_1 \Delta t P_1(t) + o(t) \quad k = 0 \end{aligned} \tag{3.22}$$

where λ_k is the birth rate (or arrival rate) when the population (number in the system) is k, and μ_k is the death rate (or departure rate) when the population is k.

3.2.1.5. Queuing Theory

Queuing theory is the study and analysis of the behaviour of queuing systems. These occur when arrivals put a demand on a finite capacity resource. Key metrics of queuing systems used for performance analysis are the mean waiting time, mean queue length or length of idle period. We describe queuing systems in terms of an *arrival rate* of customers, a *service rate* of some service facility associated with the queue and the queue itself (its length is of particular interest).

Arrivals at a service may have to wait depending on the presence of other arrivals. This waiting based on the other arrivals is known as congestion and is studied with the aid of queuing theory. Normally we treat the arrival stream as having a fixed and known stochastic structure and the service facility as being of some described design or method of operation.

Each customer arriving has a service requirement and on completion of that service departs. The total number of customers *in the system* is the number that have arrived but not yet departed. While in the system a customer can be either *in service* or delayed *in queue*. The arrival and service properties of a system will determine all other properties. Generally we can calculate measures of performance for the system in terms of averages over either *time* or *customers*. Thus we have properties of the systems such as:

- L, the mean number of customers in the system.
- λ, the arrival rate of the customers.
- w, the mean waiting time in system
- Q, the mean number of customers in queue.
- d, the mean delay in queue.

- L_s, the mean number of customers in service.
- $s = 1/\mu$, the mean time in service where μ is the service rate.

The basic relation for queuing systems is Little's law, $L=\lambda w$. In practice we often find either L or w from a corresponding limiting distribution. There are also corollaries of Little's law for delay,

$Q = \lambda d$ (3.23)

And load,

$L_s = \lambda/\mu = \rho$, the load on the system (3.24)

Considering the system to consist of regenerative processes, with each state of the system corresponding to a number of customers being in-system then by finding the proportion of time the system spends in each state we can determine this limit as time goes to infinity. Little's law is defined in terms of the means of distributions. Sometimes changes in operation of the queue will not affect these values but will effect the distributions themselves hence our interest in these system properties.

In addition to the queue itself there are many types of queuing systems that deal with different server configurations and how those servers handle service requests. The most basic case is that of a single-server queue where a single server handles all requests from a single queue. There are also models developed for multiple servers handling requests from a single queue or multiple queues each with their own queue. These are generally referred to as *queuing networks*.

We use the notation A/B/c to classify common single and multiple channel queuing models, where c is the number of channels, A denotes the inter-arrival time distribution and B denotes the service time distribution. The symbols in the A and B positions denote specific families of distributions where M denotes Poisson (exponential), D constant, G general, E_k *k*-Erlang and H hyper-exponential. FIFO behaviour and if c > 1 that all channels are fed by a single queue are assumed. Thus a single-channel queue with Poisson arrivals and exponential service is denoted M/M/1.

M/M/1 Queues

The M/M/1 queue is widely used in the modelling of telecommunications networks. Both the arrival and service rate for this type of queue are independent of both time and the state of the arrival and service processes. We can easily calculate a number of properties in this type of queue.

$\overline{N}$, The mean number of requests in the queue, $= \dfrac{\rho}{1-\rho}$ (3.25)

σ_N^2, The variance of the mean number of requests, $= \dfrac{\rho}{(1-\rho)^2}$ (3.26)

T, The average time spent in the system, $= \dfrac{1/\mu}{1-\rho}$ (3.27)

M/G/1 Queues

Up to this point we have discussed queuing systems that are entirely Markovian in nature which leads to several simplifications to their tractability. Although Markovian systems are very useful for a wide range of real-world problems they do have limitations and here we introduce a more general form, the M/G/1 queue. This has many practical applications and due to the semi-Markovian nature of the system has a relatively accessible solution for its equilibrium behaviour.

The M/G/1 queue is a single server system with Poisson arrivals and arbitrary service time distribution. The work of Pollaczek and Khiinchin allows us to calculate the mean waiting time for all customers (w) in such a system from just the offered load (ρ) and the first two moments of the service time distribution as follows:

$$w = \frac{\rho s}{2(1-\rho)}\varepsilon \qquad (3.28)$$

Where s is the mean time in service and ε is the Palm form factor of the service time distribution that is defined in terms of the first two moments as follows:

$$\varepsilon = 1 + \left(\frac{\sigma}{s}\right)^2 \quad \geq 1 \qquad (3.29)$$

The form factor is simply a dimensionless measure of the irregularity of the service time distribution. This formula for mean waiting time is called by some [ITUTele] the most important formula in teletraffic theory as it captures a wide range of complex (arbitrary) service time distributions in a form that is readily captured by experimental observation.

Telecommunications systems can often be characterised as networks of interconnected queues. Unfortunately exact solutions are only known for a limited range of queuing networks. An important consequence of the Poisson nature of arrivals at an M/M/1 queue is that in equilibrium the output of such a queue will also be a Poisson process. This allows certain types of networks of M/M/1 queues to be described by the formulae above. This is not true for queues in general. More complex problems must generally be solved through the use of approximation methods such as mean value analysis, decomposition or aggregation.

3.2.2. Fixed Point Methods

Fixed point methods are a well-known family of techniques for finding the zeros (roots) or solving a system of non-linear equations. In general these systems of equations (*n* real functions of *n* real variables) can be considered to be descriptions of transformations. For example given a system of *n* equations:

$$f_i(x^1,\ldots,x^n) = 0, \quad i = 1,\ldots,n \tag{3.30}$$

Where the superscripts denote the components of the vectors *x*. Then these *n* equations describe a transformation depending on the definition of the function $f: E \to F$, its domain *E*, and range *F*.

Usually it is not possible to explicitly determine a zero, ξ, of *f* in a finite number of steps so approximation methods are used. In general these approximation methods are iterative and start with an initial value x_0. An iteration function, $\Phi: E \to E$, is then used to compute successive approximates $x_i, i = 1,2,\ldots,\xi$. i.e.

$$x_{i+1} := \Phi(x_i), \quad i = 0,1,2,\ldots \tag{3.31}$$

If ξ is a fixed point of Φ then $\Phi(\xi) = \xi$. If we can choose our iteration function so that all its fixed points are also zeros of *f* and it is continuous in the neighbourhood of all of its fixed points, then each limit point of the sequence x_i , i = 1, 2, …., is a fixed point of Φ and hence a zero of *f*. The Newton Raphson method provides us with a general approach to defining iteration functions with these properties.

3.2.2.1. The Newton Raphson Method

If ξ is the zero of a function *f* and if *f* is differentiable in a neighbourhood *N*(ξ) of this point, then the Taylor series expansion of f about $x_o \in N(\xi)$ is

$$f(\xi) = 0 = f(x_o) + (\xi - x_0) f'(x_0) + O((\xi - x_0)^2) \tag{3.32}$$

By ignoring the higher powers of (ξ - x_0) we arrive at equations which express the point ξ approximately in terms of a given, nearby point x_0 i.e.

$$\bar{\xi} = x_0 - \frac{f(x_0)}{f'(x_0)} \tag{3.33}$$

In general $\bar{\xi}$ is merely close to the desired zero. Hence we now have our desired iteration method where:

$$x_{i+1} = \Phi(x_i), \quad \Phi(x) = x - \frac{f(x)}{f'(x)} \tag{3.34}$$

To solve this system we linearise *f*. If we assume that $x = \xi$ is a zero of *f,* that x_0 is an approximation to ξ and that *f* is differentiable for $x = x_0$ then to a first approximation

$$0 = f(\xi) \approx f(x_0) + J(\xi - x_0) \tag{3.35}$$

Where *J* is the Jacobian matrix of partial derivatives and (ξ - x_0) is the vector of differences between the initial approximations to ξ for each of the *n* equations in the system. If the Jacobian is non-singular then we can solve the equation,

$$f(x_0) + J(x_1 - x_0) = 0 \tag{3.36}$$

for x_1, an approximation that is closer than x_0 to the zero ξ. In practice (3.7) is rewritten in matrix notation as

$$\mathbf{J} \cdot \delta\mathbf{x} = -\mathbf{F} \tag{3.37}$$

Where $\delta\mathbf{x}$ is the vector of differences between the initial approximations to ξ for each of the *n* equations in the system. This can be solved by LU decomposition. This is the process of decomposing the matrices into two triangular matrices L, which only contains elements in the lower triangular, and U, which only contains elements in the upper triangular. This breaks up one linear set of equations into two successive ones which have much more trivial solutions obtained by forward and back substitution respectively. This produces a solution vector $\delta\mathbf{x}$, which is a series of corrections to our current approximation of the solution to our system of non-linear equations. Hence

$$\mathbf{x}_{new} = \mathbf{x}_{old} + \delta\mathbf{x} \tag{3.38}$$

This process is then iterated to convergence as limited by machine accuracy. This method gives quadratic convergence for a system with only a single non-linear equation or where we have an initial approximation that is sufficiently close to the zeros of the system. However, for other cases it can easily fail to converge. Hence we consider modifications to this method below.

3.2.2.2. Globally Convergent Modified Newton-Raphson Method

The convergence conditions for the Newton-Raphson Method can be greatly expanded to cover a large class of functions from arbitrary initial conditions by introducing the following modification to the method. Instead of (3.9) we use

$$\mathbf{x}_{new} = \mathbf{x}_{old} + \lambda\delta\mathbf{x}, \qquad 0 < \lambda \leq 1 \tag{3.39}$$

Where λ is chosen so that the sequence $\{h(x_i)\}$, $h(x) = f(x)^T f(x)$, is strictly monotone decreasing and the x_i converge to a minimum point of $h(x)$. Every local minimum point of *h* which satisfies $h(x) = 0$ is also

a global minimum of *h* as well as a zero of *f*. We also observe that evaluation of (3.8) defines a search (or descent) direction for *f*. This is important because it means that it is not necessary to exactly find the minimum point of *f* at each step. Instead we can rely on the fact that (3.8) will indicate the correct direction to move along *f*, thus we can simply scale back λ until we have an acceptable step. An acceptable step is defined as one in which the value of *f* is reduced. This is very useful because precisely finding the minimum of *h* at each step would be computationally very expensive.

There are still problems with this approach as the definition of an acceptable step must be tightened to ensure convergence in a timely manner. There are two issues with an acceptability definition of $f(\mathbf{x}_{new}) < f(\mathbf{x}_{old})$. Firstly, it is possible that each successive step is smaller and hence no real progress towards the minimum is made. Secondly it is possible that the step rates are too small compared with the initial rate of decrease of *f*. The first issue can be solved by requiring the average rate of decrease of *f* to be at least some fraction of the initial rate of decrease. The second can be avoided by specifying a minimum step length. In practice we find that at each step either we are sufficiently close to our zero to use a value of $\lambda = 1$ (the Newton Raphson method) or we use a value of λ between 0.5 and 0.1 our original λ. We estimate the required value of λ by modelling *f* locally as a quadratic or cubic and choosing the value of λ that minimises our model.

3.2.3. Simulation-based Network Modelling

Models are system descriptions that eliminate non-essential detail to capture the essence of the system being studied. Under the simulation methodology we study a system by observing the response of our model to artificially generated inputs. Construction of simulation models involves more design decisions because although arbitrarily complex systems can be simulated, there is always a trade-off between model complexity and execution time. In general, simulation models don't give the same degree of insight into system behaviour that mathematical analysis brings.

Simulation models are normally written in a computer programming language for a specific simulation environment, which provides support for standard simulation elements such as the generation of stochastic variables, monitoring the passage of time within the simulation and other simulation support functions such as data collection.

Each simulation run is a statistical experiment in which the response of a logical model to artificial input data is observed. This differs from normal statistical experiments because the experimenter has more control of the input data and the experimental environment. There are a number of statistical questions that inevitably occur when discussing the results of a simulation experiment. How long does it take for the system to reach statistical equilibrium? No real system run for a finite time can ever reach this; it can only be approached asymptotically. How many samples of a system variable are necessary to

ensure that the observed values are valid? To address this issue we generally construct a *confidence interval* whose endpoints are calculated from the simulation output.

There are two main ways to construct simulation of a communications system, discrete time advance or discrete event advance. In discrete time advance the simulator moves from time instant i to time instant $i + 1$ regardless of whether the system state has changed in the elapsed simulation time. In discrete event simulation the simulator clock is advanced to the next time for which there is a state change in the system e.g. a packet departs from a queue. Discrete event-based systems are generally easier to build and execute faster for periods where there is little system activity.

Generation of stochastic variables is of great importance to simulation environments. A variable of any distribution can be produced from a random variable uniformly distributed on the range [0, 1]. In practice, pseudo-random number generators are used which do not exhibit any correlation between the generated numbers. These generators are optimised for speed and when given the same seed value will always produce the same sequence of numbers. This can be important for debugging but can also cause inadvertent errors. An important property of a generator is that it has a long period before it will start to repeat the sequence of numbers produced. This is to prevent artificial correlations between infrequent events being reported.

As with any software engineering project, validation of simulation models is a required part of their development process. This becomes increasingly difficult as the complexity of the model increases and therefore places a practical limit on complexity. Typically, simulation models are validated by comparison with analytic results, the output of other simulation models or real-world experimental data.

All of the models described in this thesis were created in the OPNET™ [Opnet] simulation model development environment. This is an integrated software application designed specifically for the modelling, analysis and design of communications networks.

Chapter 4. A Middleware-based Architecture for Distributed Intelligent Networks (DIN)

In this chapter a new architecture for Intelligent Networks based on CORBA middleware technology is described. The architecture supports both entirely CORBA-based IN systems and inter-working between CORBA-based IN systems and traditional IN nodes. The behavioural and specification mapping is based on the Transactions Capabilities (TC or TCAP) layer of the SS.7 protocol stack which carries INAP as a TC-User protocol. The architecture includes the interfaces necessary to build gateways between CORBA-based and traditional SS.7-based TC-Users, algorithms for the translation of traditional TC-User specifications (e.g. ITU-T CS-1 INAP) into CORBA-based equivalents and specifications for the interaction semantics of CORBA-based IN entities.

The structure of this chapter is as follows: Section 4.1 provides the high level architectural overview, Section 4.2 provides a description of the specification translation of a TC-User specification to corresponding IDL interfaces; Section 4.3 describes the specification of the dynamic behaviour of TC-User objects in the CORBA domain, Section 4.4 describes the TC PDU oriented interfaces which provide a platform and language neutral method of interfacing to a TC/SS.7 protocol stack. In section 4.5 there is a discussion of deployment scenarios and section 4.6 provides information on implementation experiences with the specification. Finally, section 4.7 gives a summary of the chapter and some conclusions.

4.1. Architectural Overview

In Chapter 2 the prevalence of CORBA in modern telecommunications networks was identified. However this has primarily been associated with management interfaces or, more recently, third party service integration. In this Chapter we define an architecture for the development of CORBA-based IN service execution functions such as SCFs and SSFs. This architecture is structured so that it naturally supports the development of both native CORBA solutions and combined CORBA/traditional IN service networks linked by TC/CORBA gateways. The architecture is also not limited to the use of the INAP protocol (of any version) but can also support any TC/ROS-User protocol specified in ASN.1. This enables combined middleware service platform solutions to be built for IN, GSM-MAP, CAMEL, X.500 or other TC-User services. Figure 4.1 illustrates the context of our work, the DIN, in the service

layer in a general CORBA-based telecommunications service framework.

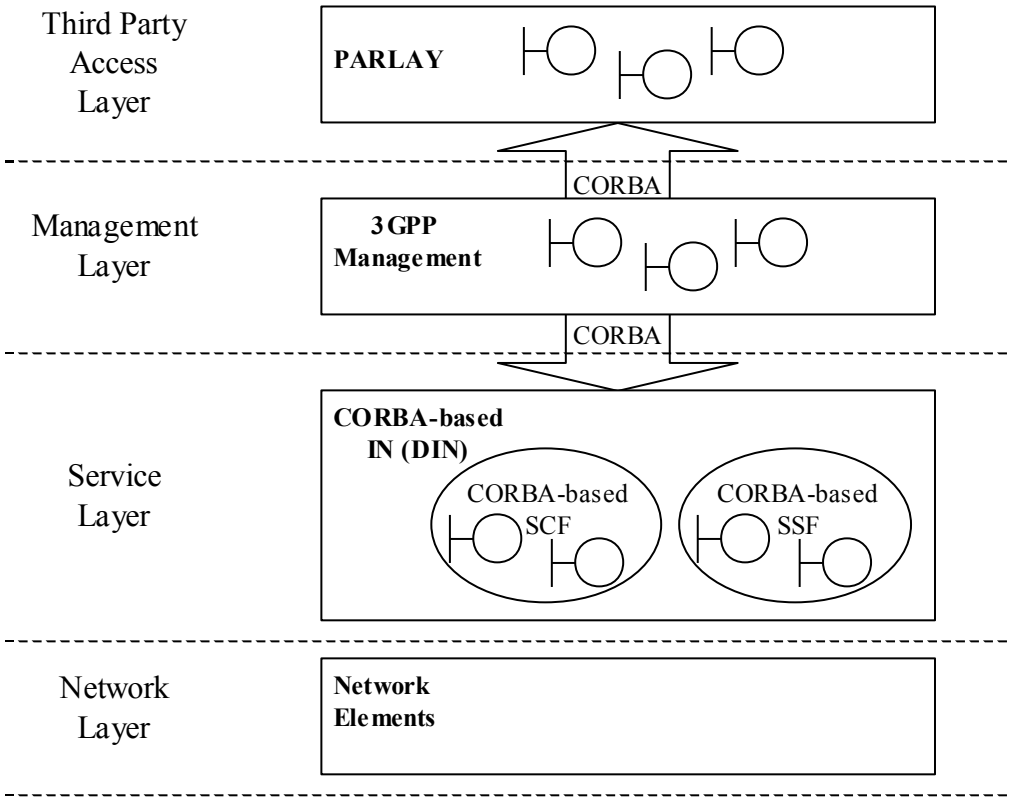


Figure 4.1: General CORBA-based Telecommunications Service Framework

The architecture we specify enables the creation of IN-based systems (DINs) that define their interfaces in OMG IDL based on INAP (or other TC-User protocols), their interaction semantics in terms of ITU-T TC interaction semantics and use the ORB for communication. This ensures that CORBA-based DIN functional entities such as SSFs or SCFs have the property of location transparency with respect to other CORBA-based DIN entities or traditional IN nodes on the SS.7 network (through the use of proxy objects at a TC-CORBA gateway). Thus this architecture supports both the migration and interworking scenarios identified by Eurescom project P508.

To achieve syntactic and semantic equivalence in the CORBA and traditional IN domains requires a mapping between existing IN-based system interfaces (based on INAP and TC) and CORBA object models and services. This mapping is split into Specification Translation (ST) and Interaction Translation (IT).

Earlier work by Mazumdar *et al.* [MAZ] on IN-CORBA interactions had proposed a loose and incomplete mapping between the two request-reply interaction paradigms: the OSI Remote Operations Service (ROS) [X880], which forms the basis for the Signaling System No. 7 (SS7) protocol called Transaction Capabilities (TC) [Q771], and the Common Object Request Broker Architecture (CORBA)

[CORBA] specified by the Object Management Group (OMG). The key new work presented here is to elaborate on those original ideas to provide a viable specification for the construction of CORBA-based IN nodes. This required:

- a revised and expanded algorithm for translating TC-User specifications into CORBA specifications (specification translation);
- a new approach to handling the interaction semantics of CORBA-based TC-User to TC-User communication (interaction translation);
- an entirely new set of interfaces, used at a gateway, to enable communication between a CORBA entity and a SS.7 protocol stack implementation.

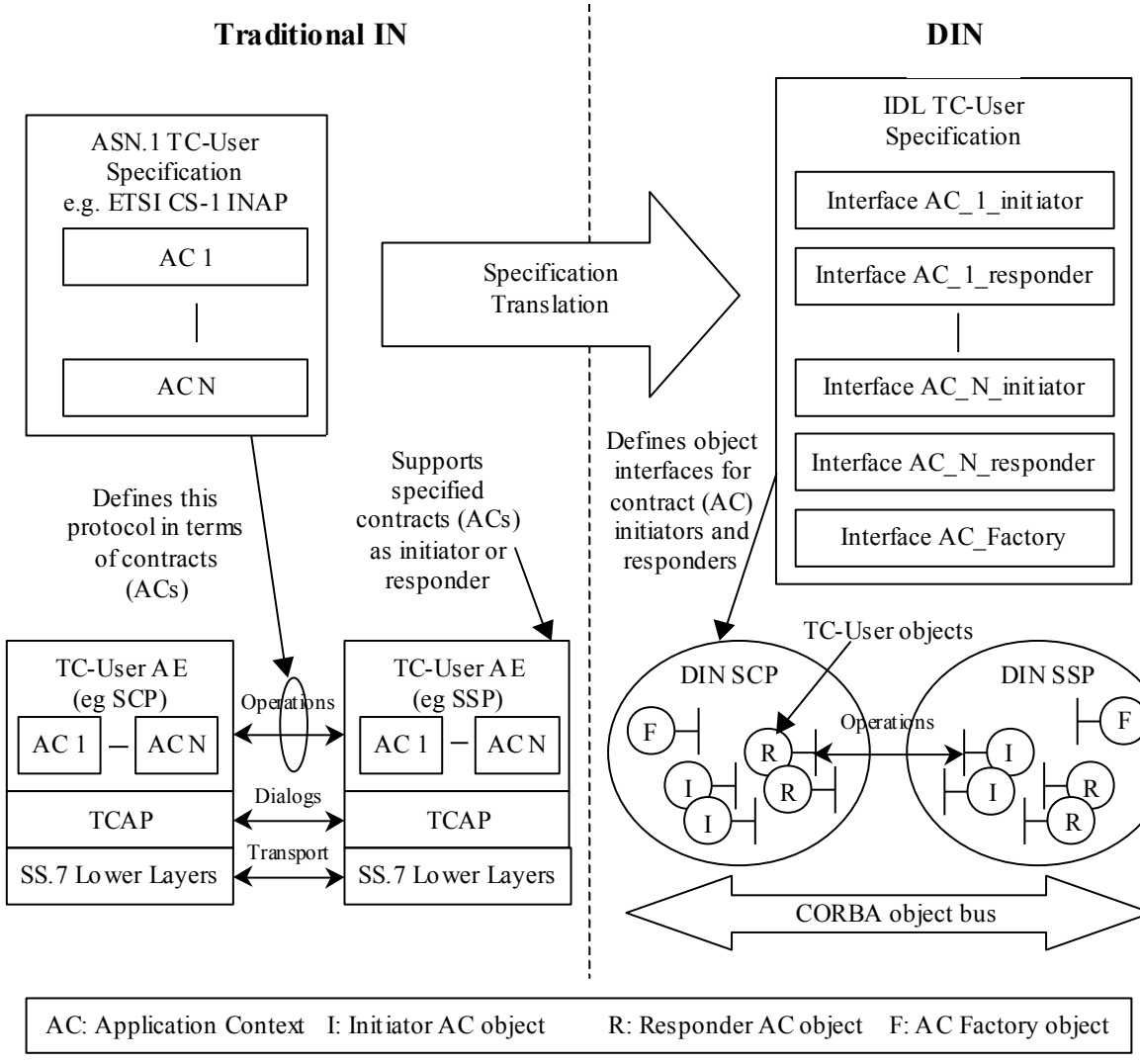


Figure 4.1: Specification translation approach and consequences for physical distribution

The general approach for specification translation (illustrated in figure 4.2) has been to:

- define a set of basic capabilities for all CORBA-based TC-Users and group these into a base class

(TcUser) from which all translated entities inherit.

- create two IDL interfaces (CORBA objects) for each application context (a defined subset of the functionality of an IN functional entity, for example the service management functionality of a SCF) specified in the ASN.1 protocol definition. Note that in post 1994 ASN.1 specifications an application context is referred to as a contract. One of the generated interfaces represents the initiator of a TC communication session within this application context and the other the responder. Normally a given functional entity would only implement one of these interfaces and its peer would implement the other.
- create one IDL interface for a factory entity to act as a manager of instances of the application context interfaces (objects)

This defines the granularity of distribution for a given functional entity for the purposes of CORBA. For example an ETSI CS-1 SCF would consist of eight collaborating (distributable) objects. Thus, by moving to a DIN solution we have changed the focus of specification from defining a protocol between computational nodes (the traditional approach) to defining discrete clumps of IN service related functionality which are capable of arbitrary distribution or replication on the many computational nodes that make up our CORBA platform.

TC-based signalling is key to most telecommunications network-based services, hence the assumptions of this specification are general enough to allow inter-working for *all* TC-based applications, and not just the most popular one, namely that for Intelligent Networks (IN). Thus this architecture enables the creation of hybrid INAP/GSM-MAP nodes and eases integration with CORBA-based management initiatives such as OMG CORBA/TMN Interworking, 3GPP service management specifications, Parlay third party service access and experimental CORBA-based service creation environments. Finally, by eliminating the expensive SS.7 network from practical network service solutions it is hoped that new lightweight service providers will be enabled.

4.1.1. Basic Concepts

The first major attempt to translate ITU/OSI protocols and models to the CORBA domain was undertaken by the TMF-X/Open Joint Inter-Domain Management task force who worked on TMN migration to CORBA [JIDM] (henceforth referred to as JIDM). Their work identified two key aspects of the mapping process:

- **Specification Translation:** Defined as specification of the algorithms that enable the static translation of specifications to/from ITU/ISO Abstract Syntax Notation 1 (ASN.1) [X680] from/to OMG Interface Definition Language (IDL) [CORBA]. Our work has extended the JIDM specification translation algorithms to cater for five new classes of ASN.1 constructs related to application contexts that are required for TC-User protocol definition.

- **Interaction Translation:** Defined as specification of the IDL interfaces and semantics to support the runtime translation of the basic interactions between systems, for example definition of the IDL interfaces to support the naming and addressing conventions of SS.7 in a CORBA environment. None of the JIDM interaction translation mechanisms were suitable for TC-User applications and so an entirely new interaction translation approach had to be designed.

Some interaction translation mechanisms are only used in a gateway environment. However the basic methodology of the mapping requires that interactions between TC-User entities in the CORBA domain are indistinguishable from interactions between CORBA-based TC-Users and traditional SS.7-based TC-Users via proxy objects at gateways. Figure 4.3 illustrates how these concepts are combined with OMG standardised implementation language mappings to develop a set of CORBA-based TC-User applications, for example a CORBA-based SCP and SSP.

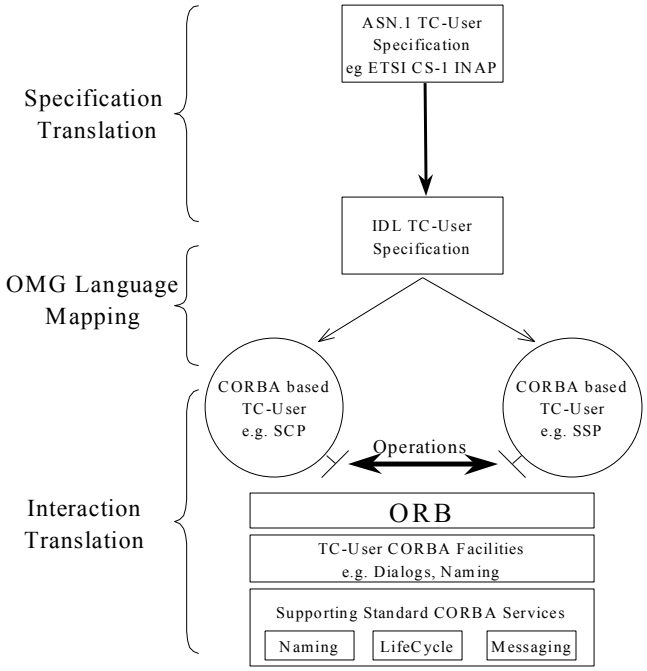


Figure 4.3: Basic Concepts for CORBA/TC Mapping

The scenario illustrated in figure 4.3 also demonstrates another advantage of the CORBA-based approach, the existence of standardised implementation language mappings. In the world of ASN.1 specifications there are no standardised language mappings and hence application developers typically have to purchase or develop special encoder/decoder libraries to map between protocol bits "on the wire" and an easily accessible version held in local memory. Each new protocol specified in ASN.1 requires new libraries. For CORBA-based implementations, a new IDL specification is simply processed by a tool supplied by the ORB-vendor. This generates implementation code (stubs and

skeletons) according to the OMG standardised language mapping.

The ST we define is an extension of the JIDM specification translation standard already adopted by The Open Group/NMF and also adopted by the *OMG TMN/CORBA Interworking* specification. The JIDM work uses TMN's GDMO templates to define IDL interfaces for managed objects. In the TC-User domain there are no GDMO templates defined, instead TC-User protocols are defined by several ASN.1 constructs, either using the ASN.1 macro notation (pre-1994) or ASN.1 information object classes (post-1994). The JIDM approach is expanded here to define mappings for these constructs to IDL interfaces. The JIDM translation algorithms of basic ASN.1 types to IDL are retained. As the JIDM ST does not provide a mapping of the ROS constructs, there will be no discrepancy between the two ST algorithms. Specifically, the ST will map a generic TC-User protocol (defined using Abstract Syntax Notation One, and using the ROS Information Object Classes **`CONTRACT, OPERATION-PACKAGE, OPERATION`**, **`ERROR`** and **`EXTENSION`** to specify remote operations) to the corresponding CORBA IDL constructs. Our JIDM ST extensions must also take into account the TC interaction semantics we wish to convey within the CORBA domain, for example by including special additional parameters in operations to carry this information which is not embedded in normal ORB application interaction semantics.

The IT we define specifies how to locate, name, and interact with TC-User implementations in the CORBA domain. This completely replaces the earlier work of Mazumdar *et al.* and was never part of the JIDM specifications that did not address TC-users. It includes defining the mapping of the CORBA TC-User dialog handling facilities onto appropriate behaviour at the gateway such that a TC-User implemented as a CORBA object has access to these TC facilities when communicating with a traditional IN node through a gateway. Special attention has been paid to addressing potential performance issues with regard to these dialog-handling facilities. We also include a specification of naming support for both TC-User protocols specified in ASN.1 macro notation (pre-1994) or ASN.1 information object classes (post-1994). A range of generic CORBA-based TC-User object "helper" interfaces are also defined to simplify configuration and interaction. Finally there are two interfaces provided that will only be used in TC/CORBA gateway implementations to simplify administration and ASN.1 to IDL mappings.

One further new facility is provided in our architecture, namely IDL interfaces to the TC protocol stack (hereafter called the TC PDU-oriented interfaces), which allow proxy objects standardised, efficient access to the TC/SS7 protocol stack. This is the first time that any standardised API has been proposed for the SS.7 TC layer.

4.1.2. CORBA/TC Inter-working Gateway

An inter-working gateway acts as a bridge between two domains, mapping the differences (if such exist) in message format, addressing and interaction paradigms. In our case, the so-called "legacy" domain is that of nodes in a SS.7 network communicating using the SS.7 TC/SCCP/MTP protocol suite with nodes implemented using CORBA technology. Figure 4.4 illustrates two scenarios for inter-working in this manner, although it is very likely that the case of CORBA-based Service Control Points (SCP) inter-working with the considerably larger base of "legacy" IN-capable switches, the Service Switching Points (SSP), are going to dominate in actual deployment. This corresponds to the vision of inter-working as described by Eurescom project P508 [P508].

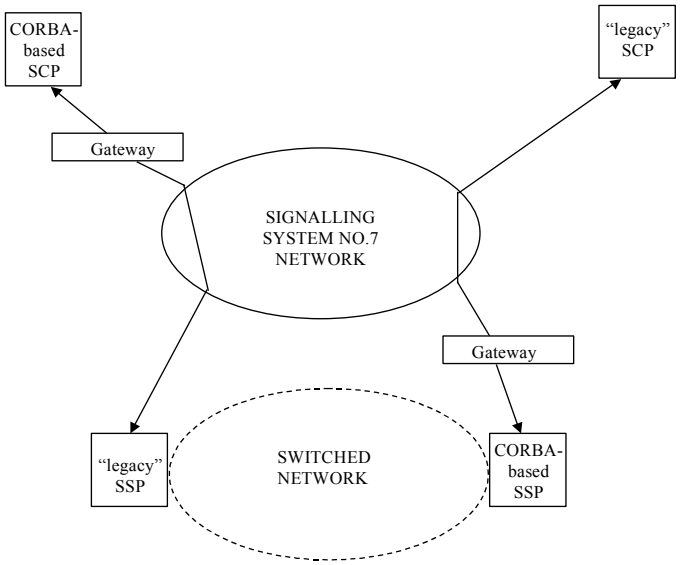


Figure 4.4: Inter-working Scenarios

The high-level gateway architecture is illustrated in figure 4.5. This expands on the basic concepts shown in figure 4.4 to include a TC-User outside the CORBA domain. Note that from the point of view of the CORBA-based TC-User, a SCP in the figure, its interactions with the SSP proxy object in the gateway are identical to those it would have with a CORBA-based SSP. This preserves the concept of *domain transparency* that we define as a primary objective of our work. Similarly the SS.7-based SSP should be able to interact with the CORBA-based SCP as if it was in the SS.7 domain

The right hand side of the figure, marked Legacy Domain, shows the typical structure of a TC-User, in this case that of a SSP which can support several Application Contexts (AC) each of which is made up of combinations of Application Service Elements (ASE). This TC-User communicates with a SCP, which makes up the remainder of the figure, using the connectionless transport capabilities of the SS7 network, namely the Signalling Connection Control Part (SCCP) and Message Transfer Part (MTP).

The legacy domain terminates at the TC stack interface to a TC User, shown in the figure as the "TCU PDU oriented interface". This is a set of IDL interfaces designed to encapsulate any vendor's SS.7 stack at the TC level. It is only used in gateway nodes.

At this point, of course, the software architecture is implementation/vendor-specific. However, from the point of view of a CORBA-based implementation of a TC User, such as an SCP, one can take advantage of CORBA's separation of interfaces to objects from the implementation of these interfaces to define the basic set of interacting objects which provide a CORBA-centric view of the entire system. This is done by defining a "proxy" object that represents the external or "legacy" domain, namely the SSP. Thus interactions between the CORBA-based SCP and the "proxy" SSP follow the distributed object interaction paradigm, making use of various CORBA infrastructure services like Naming, LifeCycle and Messaging. As the TC interaction paradigm adds some additional features, such as an association between two TC Users called a dialog (or transaction), and TC User entities are named somewhat differently, it is necessary to enhance the CORBA services with some additional interfaces to include these TC-specific features. Subsequent sections will elaborate on these enhancements.

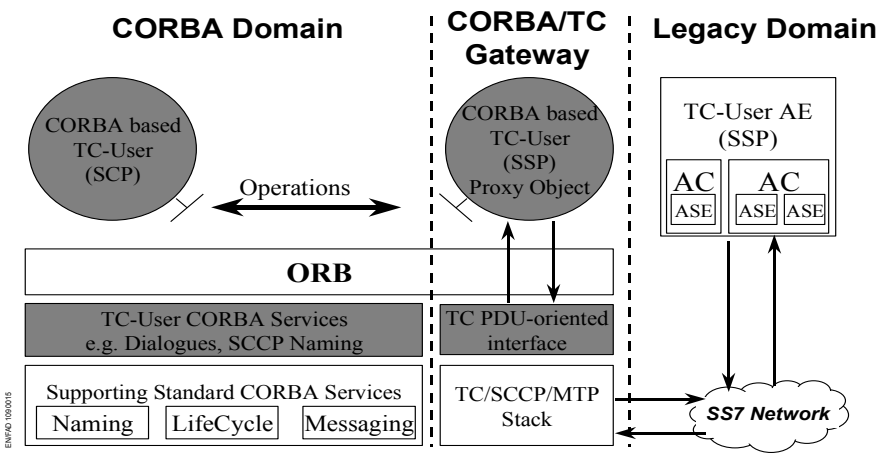


Figure 4.5: High-level Architecture of CORBA/TC Gateway

The gateway performs several functions: it contains information and procedures on how to locate and interact with TC-User implementations in the CORBA domain; it has to map TC dialog handling facilities (i.e., the sending and receiving of TC messages such as BEGIN, CONTINUE, etc.) in a manner consistent with the expected external interface to the legacy domain; and it has to convert CORBA calls marshaled used the CORBA General Inter-ORB Protocol (GIOP) and Common Data Representation (CDR) into TC Application Protocol Data Units (APDUs) encoded using Basic Encoding Rules (BER).

4.2. Specification Translation

Specification Translation (ST) defines a set of algorithms for mapping from the Abstract Syntax One (ASN.1) notation description of TC-Users into Interface Definition Language (IDL) interfaces to CORBA objects representing these TC-Users. TC User interactions are defined in terms of operations (defined by the ASN.1 construct `OPERATION`) and potential errors (defined by the ASN.1 construct `ERROR`). Groups of related operations are packaged using the construct `OPERATION-PACKAGE`, while sets of operation packages form a contract between two interacting TC-Users (defined by the construct `CONTRACT`). The ST defined our specification is based upon the ST algorithm [JIDM] defined jointly NMF and The Open Group for mapping GDMO templates to IDL interfaces for managed objects. While our work uses the same mapping for basic ASN.1 types as defined by [JIDM], the above-mentioned ASN.1 constructs used in TC were not required by the GDMO mappings and hence not defined in that work; so the CORBA/TC inter-working specification provides these additional mapping algorithms. In addition to these constructs which are defined in the 1994 version of the ROS specification, there are earlier analogous constructs using the generic `MACRO` construct of ASN.1 defined in the 1988 version of ASN.1. These cases are also handled, as they are important for ITU/ETSI IN Capability Set 1 [ETSI] and the general applicability of the approach.

4.2.1. A Specification Translation Example

The ST is best illustrated through an example. Table 4.1 provides an ASN.1 specification describing a simple (but hypothetical) query/response service for a telephone directory lookup. One TC User, representing the user, invokes the operation `getNumber` by providing the `Name` of the person whose `TelephoneNumber` is returned as a result of performing the operation. However there may be instances when, before completing the operation, the other TC-User — the "directory" — may need further clarifying information (such as a person's middle initial or a postal code so as to be able to narrow down the choice of alternatives), which it does by invoking a linked operation `clarify`. There is no straightforward notation for specifying which TC PDUs (BEGIN, CONTINUE, etc.) message should be used to carry these operations; so the PDUs are indicated as ASN.1 comments. In this example, the `getNumber` operation is obviously sent in a BEGIN, while the response, whether a reply to this operation, and error, or the linked invocation `clarify`, is returned in a CONTINUE, the understanding being that it is up to the initiator to END the dialog.

ASN.1 Specification

```
TelDirLookupProtocol{objid1} DEFINITIONS ::=
BEGIN

IMPORTS OPERATION, ERROR FROM Remote-Operations-Information-Objects {joint-
```

```
iso-itu-t remote-operations(4) informationObjects(5) version1(0)};

getNumber OPERATION ::=
{
     ARGUMENT          Name
     RESULT            TelephoneNumber
     ERRORS            {unknown | noAccess}
     LINKED            {clarify}
     CODE              local:1
}
-- sent in a BEGIN message; returns in a CONTINUE message
-- timer = 5 seconds

clarify OPERATION ::=
{
     ARGUMENT          Clarification
     RESULT            RequestedInformation  OPTIONAL TRUE
     ERRORS            {unknown}
     CODE              local:2
}
-- sent in a CONTINUE message; returns in a CONTINUE message
-- timer = 60 seconds

unknown ERROR ::=
{
     PARAMETER     Cause
     CODE          local:1
}
noAccess ERROR ::=
{
     CODE     local: 2
}
--errors returned in a CONTINUE message

Name ::= VisibleString

TelephoneNumber ::= SEQUENCE SIZE(10) OF INTEGER (0..9)

Clarification ::= ENUMERATED{provideZipCode(0), provideMiddleInitial(1),...}

RequestedInformation ::= CHOICE
          {
          zipCode       [0] IMPLICIT SEQUENCE SIZE(5) OF INTEGER(0..9),
          middleInitial [1] VisibleString
          }

Cause ::= ENUMERATED{noSuchName(0), noSuchZipCode(1), ...}

END
```

Table 4.1: ASN.1 specification for a simple directory lookup protocol

We further assume that the telephone directory user is in the "legacy" domain, and communicates using SS.7/TC with the "directory" which is implemented using CORBA technology, i.e., its interactions are encapsulated using a CORBA IDL interface. Based on the discussion accompanying Figure 4.5, this means that the user is represented as a "proxy" CORBA object at the CORBA/TC gateway. The

purpose of ST is to generate the CORBA IDL interfaces for these two CORBA objects. The ASN.1 to IDL mapping that generates these interfaces for the example are given in Table 4.2.

Translated IDL Specification

```
module TelDirLookupProtocol {

// ASN.1 types are first, then interfaces, then  exceptions then operations

typedef ASN1_VisibleString NameType;

typedef sequence <ASN1_Integer, 10> TelephoneNumberType;

enum ClarificationType {provideZipCode, provideMiddleInitial};

enum RequestedInformationTypeChoice {zipCodeChoice, middleInitialChoice};

union RequestedInformationType switch( RequestedInformationTypeChoice)
{
        case zipCodeChoice: sequence<ASN1_Integer, 5> zipCode;
        case middleInitialChoice : ASN1_VisibleString middleInitial;};

enum CauseType { noSuchName, noSuchZipCode };

interface TcUserFactory:TcSignaling::TcUserGenericFactory{

        DefAc create_DefAcInitiator ( );

        DefAc create_DefAcResponder ( in DefAc initiator,
                                in TcSignaling::AssociationId a_id,
                                in TcSignaling::TcContextSetting tc_context_setting)
                raises (TcSignaling::NoMoreAssociations,
                        TcSignaling::UnsupportedTcContext);

        } //end of interface

interface DefAc:TcSignaling::TcUser{

        exception unknown { CauseType cause; DialogFlowCtr ctr; };

        exception noAccess { DialogFlowCtr ctr;};

        TelephoneNumberType getNumber ( in NameType name,
                                        inout TcContext ctext )
                raises (unknown, noAccess );

        RequestedInformationType clarify(in ClarificationType clarification,
                                        inout TcLinkedContext ctext)
                raises ( unknown );

        } //end of interface
} //end of module
```

Translated IDL Specification

Table 4.2: The ASN.1 to IDL mapping for the "Directory" example (shown in Table 4.1)

An explanation of the algorithmic translation from ASN.1 to IDL, as shown in Table 4.2, is as follows (keeping in mind that this would be generated automatically by the ASN.1-to-IDL translator and needs no human intervention):

The ASN.1 module name is translated into an identical IDL module name as described in [JIDM].

An interface called **TcUserFactory** which inherits from **TcSignaling::TcUserGenericFactory** is created which contains two operations: **create_DefAcInitiator** and **create_DefAcResponder**. This interface is used to create the initiator and responder CORBA objects. Both operations return the object reference to the interface, **DefAc**, which inherits from **TcSignaling::TcUser**, and which represents the actual TC User interactions. In our simple example, the initiator and responder interfaces are identical, but in a more complex example, these would be different. Note also that the initiator interface takes no creation parameters, and, indeed, could even have been made implicit. This is because the manner in which the initiator of an interaction is created is typically outside the scope of the interactions themselves. The responder interface, on the other hand, takes several parameters:

The object reference of the initiator interface, **DefAc**, so that the created object may participate in a two-way exchange of operations with the initiator;

An association id, **a_id**, which is a unique identifier for the TC dialog between the two objects;

A parameter, **tc_context_setting**, which permits the creation process to choose whether the created object will support either or both of the use of association ids; if this is not supported, the created TC-User/CORBA object can only support one (implicit) association;

The use of parameters in operations (see more on this below) that identify which TC PDUs are to be used/have been received at the gateway with the operation; if this is not supported, the gateway will choose some default PDU handling behavior;

This parameter is intended for the case when it is necessary for the created object to be made aware of TC semantics such as dialog handling. There may be instances when it is necessary to shield the CORBA programmer from all TC details; therefore, the submission ensures that all conforming implementations must permit no support for these features.

The parent factory interface **TcSignaling::TcUserGenericFactory**, from which the specific factory described above inherits, consists of generic operations to create TcUser object/interfaces, and is particularly suited for use with dynamic gateways where statically translated information (such as Table 2 above) may not be unavailable.

And finally, the actual interface, **DefAc**, where

IDL datatypes are **declared** for ASN.1 data types;

TC-User errors are mapped to IDL exceptions, with ASN.1 error parameter names being mapped to IDL parameters following [JIDM] naming rules, together with an additional parameter, of type **DialogFlowCtr**, indicating which TC PDU is to be used at the external interface for communicating the error to the legacy system;

TC-User operations are mapped to IDL operations with several parameters:

An **in** parameter containing the translated argument of the operation;

An **inout** parameter of type **TcContext** (or **TcLinkedContext**) providing the invoke id (and linked id) of the operation, the association id over which this invocation is occurring, and the TC PDU to be used when communicating this operation to a legacy system. Note that the submission had a requirement that the semantics of TC/ROS operations be maintained. This means that it is necessary to convey the invoke id information, even though a CORBA invocation does not need this information to correlate invocations and their returns. However, TC has the concept of "linked" operations, which requires the use of the invoke id to convey which operation an invocation is being linked to, and again this has no equivalence in CORBA. Thus, while it is possible to substantially shield the CORBA programmer from knowledge of TC by choosing to create CORBA-based TC-Users without association ids and **DialogFlowCtr** information, it is not possible to altogether remove the influence of TC and its underlying Remote Operations Service (ROS) paradigm.

An IDL result type which is either **void** (if there is no result returned) or the ASN.1 result parameter mapped according to [JIDM] to the corresponding IDL data type.

An IDL **raises** expression containing the IDL exception names for each of the errors that this operation may have.

In addition to the mapping generated in the right hand column of Table 4.3, there is a repository created which contains information that cannot be algorithmically generated such as information contained in ASN.1 comments (such as operation timer values), or for which there is no CORBA equivalent (such as TC operation and error codes). A CORBA interface, called **TcRepository**, provides access to the data contained here. Its use is illustrated later in section 4.3. The repository for our example is as follows (but please note that the tabular structure is purely illustrative):

# Scoped-Name	IDType	ID	Type	Timer
TelDirLookupProtocol::DefAc::getNumber	local	1	operation	5

TelDirLookupProtocol::DefAc::clarify	local	2	operation	60
TelDirLookupProtocol::DefAc::unknown	local	1	error	None
TelDirLookupProtocol::DefAc::noAccess	local	2	error	None

Table 4.3 Contents of the TcRepository after Specification Translation

A summary of the ST process is illustrated in Figure 4.4.

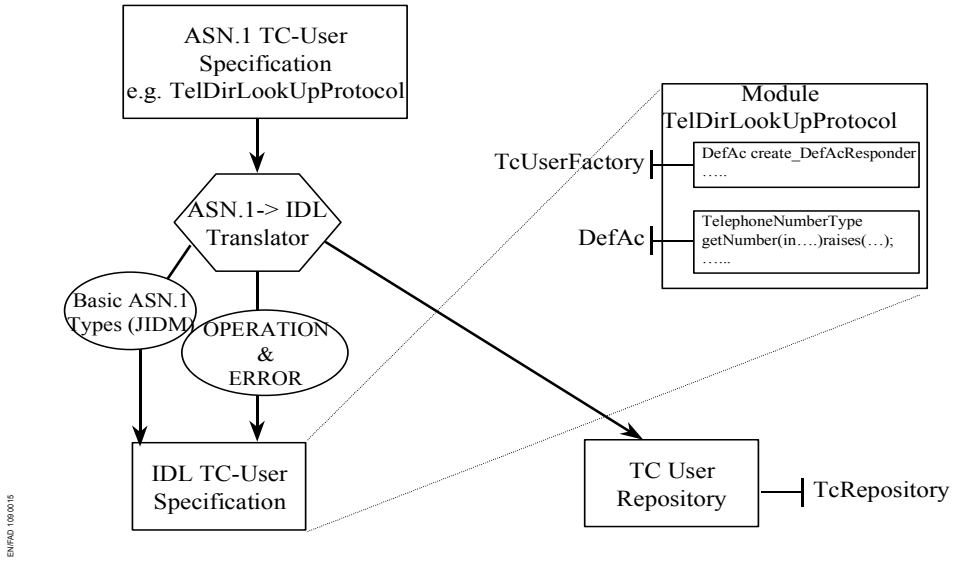


Figure 4.6: The Specification Translation Process

At this point, the programmer has to implement the two statically generated interfaces, the **TcUserFactory** and **DefAc**. **DefAc** is of course a symmetrical interface, but the implementation of its methods are not. The implementation of **DefAc** on the responder interface provides a method body for the `getName` operation, while the initiator implements only the `clarify` method of the same interface. Thus, in the case where the ASN.1 specifications do not provide a complete specification through the notation itself, as for instance happens in specifications where the Application Context (or Contract) is implicit, the single, generated IDL has also to be supported by instructions on which methods to implement on the objects which support the interface. In cases where Application Contexts (or Contracts) are specified in the ASN.1 the specification translation algorithm will generate specific, asymmetric initiator and responder IDL interfaces. Full details of the specification translation procedure are specified in Appendix 1.

4.3. Interaction Translation

Interaction Translation (IT) deals with how to name, locate and interact with TC-User implementations in the CORBA domain. These interactions are defined in such a way that the implementation difficulty of proxy objects at a gateway node is minimised. In essence, the TC-User CORBA objects have their interaction semantics defined such that they mimic normal TC interactions in the SS.7 domain. IT includes defining the mapping of the TC dialog handling facilities (i.e., the receiving and sending of TC PDUs) so that TC-User CORBA objects can be aware of, and change if needed, the state of the TC dialog. At a proxy object in a gateway this includes an implicit requirement to convert between TC APDUs encoded in BER and CORBA calls using GIOP and encoded in CDR. From a logical point of view, these functions are performed at the gateway. However wherever the specification provides an IDL interface to a capability (e.g., to a name resolution service), CORBA's location transparency allows for a physical distribution which can be based on criteria such as fault tolerance, load balancing, ease of maintenance, etc. The availability of this implementation flexibility to adapt to these non-functional requirements is a key design goal for telecommunications applications.

The following subsections will describe each of the major functions of IT in greater detail. The full details of interaction translation, including the specified IDL interface definitions, are detailed in [OMGIN].

4.3.1. TC User Objects

A set of CORBA interfaces are defined to support TC-User to TC-User interactions between either CORBA-based implementations or a mixture of CORBA-based implementations and Proxy CORBA objects (at gateways) that communicate with traditional TC-User implementations. This includes the preservation of location transparency and maximum re-use of standard CORBA Services for CORBA-based implementations. The main TC concept modelled is the dialog, which has no parallel in the CORBA domain. A TcSignaling Association is defined which may be mapped onto a TC dialog. In order to insulate CORBA objects from the details of TC dialogs:

- starting an association is constrained by the use of the CORBA Naming Service to find a suitable **TcUserFactory** interface and then invoking an appropriate **create** operation with suitable parameters. This models the operation of the TC-BEGIN dialog primitive and associated semantics.
- each association takes place between a pair of interacting CORBA objects. This does not preclude the use of a specific CORBA object in multiple, simultaneous associations as each operation within an association carries with it an **AssociationId** referring to a specific association.

To get location-independent interaction of TC-user applications, four major interaction features are supported:

1. application (TC-User) location (finding)

2. association initiation

3. association maintenance

4. operation invocation

In general, application location and association initiation are provided by the CORBA Naming Service and the Life Cycle Service, association maintenance is provided by the base interfaces of all TC-user CORBA objects and operation invocation is provided by specification translation, the ORB, the Messaging Service, and optionally the Interface Repository and TC Repository.

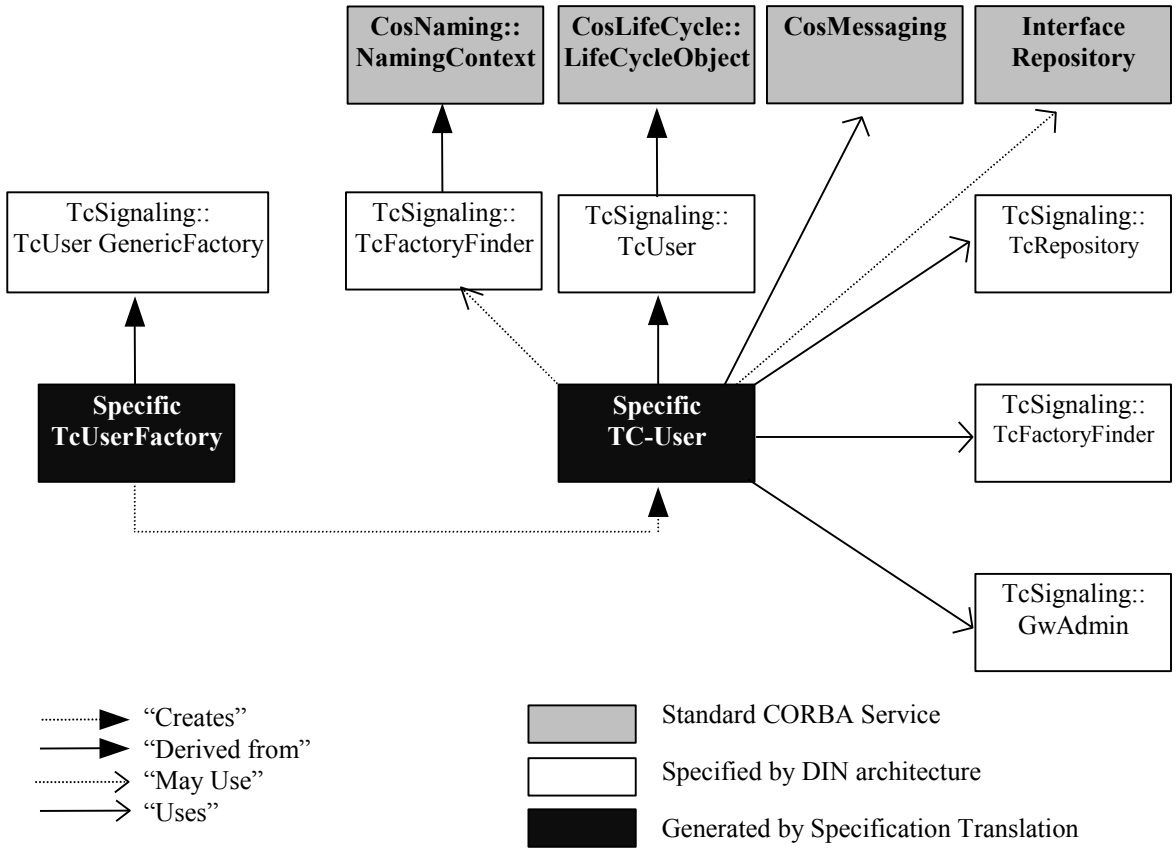


Figure 4.7: Interfaces in the DIN Architecture

The interfaces defined in this architecture are as follows.

TcUser

This interface inherits from the **COS LifeCycleObject** and defines operations that are supported by all TC-User objects to create a new association, and to end or abort an existing association. It also defines a readonly attribute that identifies if TC context setting information (e.g., to indicate TC dialog handling primitives) should be included in associations with this object.

TcFactoryFinder

This interface provides "wrapper" operations for TC-User objects to register with the **CORBA Naming Service**, an operation to find such objects, and a method to explicitly replace information in the Naming Service rather than just add new entries.

TcUserGenericFactory

This interface defines generic create operations for TC-User objects. It provides a common type for all factories generated during Specification Translation.

TcServiceFinder

This is a helper interface to find references to all the TC-User CORBA services.

TcRepository

This interface provides standardized access to the information generated during specification translation providing the mappings from TC/ROS operation, extension and error codes to IDL scoped names.

GwAdmin

This interface provides a single point of contact for accessing all gateway-related functions. In particular, it defines readonly attributes for the **TcFactoryFinder** and **TcPduProvider** interfaces.

The relationships between the various interfaces are illustrated using the class diagrams in Figure 4.7.

4.3.2. Naming and Locating Objects

Naming is the means of identifying an object (in our case the target of the TC/ROS operation) in a location-independent way. An important factor in the inter-working of CORBA-based implementations with TC/SS7 applications is the issue of harmonizing the naming aspects of the two domains. The gateway where such inter-working is to be performed will receive invocations addressed to named objects in one domain, and will have to interpret the requested target name and resolve it into an address that is recognizable in the other domain.

TC-based applications are defined as providing access to the operations of a remote object (i.e. its interface in CORBA terms) by using a communications path "directly" to the object being accessed. The communications "path" terminates at the OSI Application Entity (AE) that is the external communication interface to the object. The object itself is not named in the invocation; rather the AE associated with the object is named. Communications in OSIuses naming to allow one OSI AE establish an association with another AE. The initiating AE locates the address of its peer that it wants to "talk" to by using a name to look up the address in a directory. TC/SS.7 applications use a similar directory look-up technique. The application provides a Global Title to the "directory" — which is the

SS.7 Global Title Translation (GTT) function — which then provides the tuple = {Sub-system Number, Point Code}. It has been shown elsewhere [MIT] that, because of the absence of intervening layers in SS.7, the Sub-system Number (SSN) effectively locates an AE at a SS.7 node.

CORBA works differently. A client of an interface obtains an object reference to a server that implements that interface. The standard way to do this is to use the OMG Naming Service, which provides a name-to-object reference lookup. The name-to-object reference binding is done at the time the server object is created. It is understood that once such a name binding is advertised via the Naming Service, any client can "resolve" the name of the interface to an object reference and begin invoking operations on that target reference. Also, CORBA imposes no restriction on the naming conventions; so the naming Service can be used to support many different naming schemas (defined elsewhere).

To reconcile these two different ways of locating the target, consider the case of a TC BEGIN message received from the legacy SS.7 domain at the gateway. (Obviously the GT of the target was partially resolved through a network GTT to be able to point to the gateway.) The name tree of the CORBA Naming Service at the gateway is set up to resolve the received GT to the object reference of a factory that can create objects of the target type. (In our example from section 4.2 this would be a reference to the generated **TcUserFactory** interface.) Figure 4.8 shows the GT based naming tree for the CORBA Naming Service.

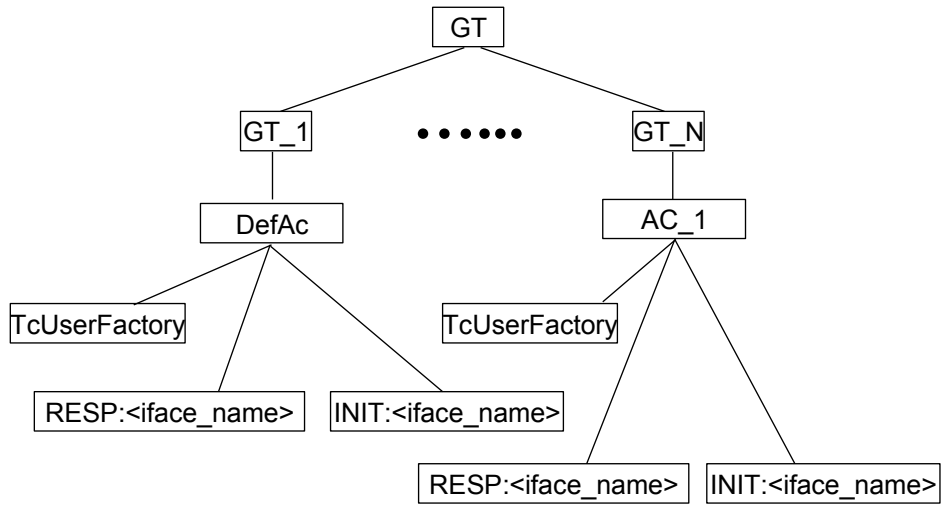


Figure 4.8: GT-based Naming Tree Using the CORBA Naming Service

The root of the GT based naming tree is named "GT" below which are naming contexts based on the

particular GTs supported by that node. The node below a particular GT is either named by the string "DefAc" (for the case, as in our example, where no explicit Application Context is indicated by the notation) or the string "AC_1" corresponding to the specific Application Context, AC_1, indicated for that TC-User. The leaves of the naming tree are nodes, which map the name "TcUserFactory" to an object reference for the corresponding factory, or Naming Contexts whose names are the IDL scoped names of the initiator and responder interfaces for objects created by that factory. The latter nodes are there so that a gateway can find the correct initiator proxy object to create for an incoming SS.7 dialog and to support the generic creation operation on the base factory interface, **TcUserGenericFactory**, usually used by dynamic implementations. When using a generic creation operation the scoped name of the interface to be created must be supplied.

Note that this naming structure permits many different types of CORBA-based TC-User objects to be resident in the CORBA domain, each accessible by the appropriate combination of a GT and (optionally) an Application Context. A CORBA-based TC-User objects or gateway may have already stored a reference to an object of the target type already (perhaps as a result of a previous request received for the same target; alternatively, it could have created a "pool" of such objects in anticipation of such requests. In any case, if the object does not exist, the obtained factory object reference allows it to create the target object, providing it with the object reference of the initiator interface. The result of the creation operation is the object reference of the responder interface.

Rather than directly manipulate the CORBA Naming Service, the submission provides a "helper" interface, **TcFactoryFinder**, which in effect encapsulates the Naming Service and allows TC-User objects to register with and also resolve names using the Naming Service or some internal mechanism such as a local database.

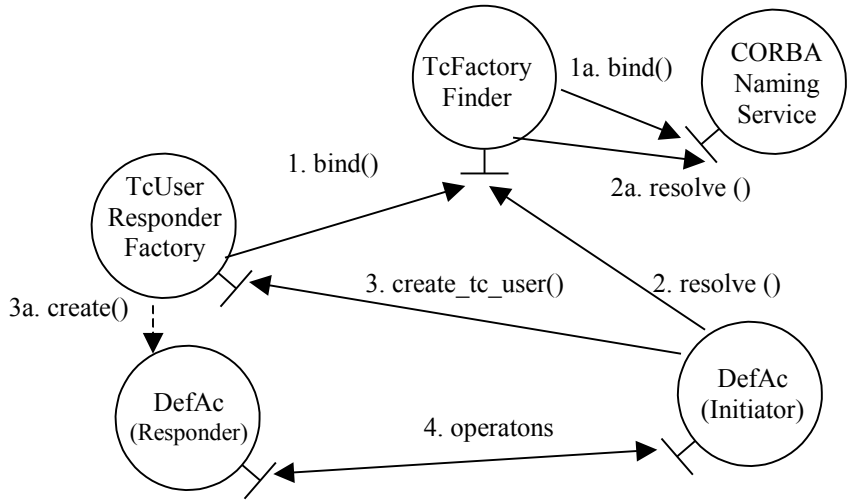


Figure 4.9: TC-User Location and Association Initiation in the CORBA Domain

Figure 4.9 illustrates the application location process for the system used as an example for specification translation. In this case we have two **DefAc** TC-User objects, a responder and an initiator, that need to locate each other in order to create an association between them (which would be mapped onto a dialog in the SS.7 domain if one of the objects was a Proxy object residing at a gateway).

A **TcUserFactory** interface binds (step 1) with a **TcFactoryFinder** object to register with a particular Global Title (GT). This registers its services as particular Application Entity in the CORBA domain. Optionally (step 1a), the **TcFactoryFinder** object may bind this information into the standard CORBA naming service or perhaps store it locally in a database (not illustrated).

Step 2 shows the initiator **DefAc** (TC-User) object attempt to locate the services of a particular Application Entity at a particular Global Title (name). This is achieved by requesting that the **TcFactoryFinder** object resolves a known GT into an object reference for a suitable TC-User factory object i.e. one that is capable of creating the type of TC-User objects that the initiator wishes to establish an association with. Optionally (step 2a), the **TcFactoryFinder** object may find this information in the standard CORBA naming service or perhaps in a local database (not illustrated).

Once a reference to the appropriate responder interface creation factory has been returned, a request (step 3) can be made to initiate a TC-User interaction (with the appropriate initiator interface). The responder interface creation factory must be supplied with an object reference to the corresponding initiator interface in the **create_tc_user** operation. Various creation parameters may be passed in the create request to constrain the created object (see next section). This initiates a new association and allows TC-like two-way communication (dialog). In the context of this association operations may now be invoked (step 4) on the TC-User responder object or the TC-User initiator object. More complex specifications that define multiple application contexts have similar behaviour but may have up to one TC-User factory defined for each application context supported by a given Application Entity.

4.3.3. Dialog State Maintenance

TC applications communicate with each other through a dialog that identifies a "pipe" over which operations may be invoked in either direction typically based on some pre-defined application-specific script. The object location mechanism described in Section 4.3.1 ensures that the two CORBA objects (one representing the initiator of the dialog and the other the responder of the dialog) have each other's object reference so that each may now invoke the operations defined on the other's interface. CORBA has no notion of a persistent relationship between two CORBA objects; so if the gateway is to retain the semantics of the TC/ROS interaction paradigm, it is necessary to provide some TC-specific semantic information to the CORBA objects.

Recall from Section 4.2 that the ST process generates the TC-User specific IDL interface(s)

corresponding to that described by the ASN.1 specification of a TC-User application. The(se) interface(s) inherit from **TcSignaling::TcUser**, which provides some operations and an attribute which are to be supported by all TC Users, namely

- An operation to begin a new association identified by the accompanying parameter **a_id** (association ID), between the initiator interface and the responder interface, with an exception to signal that no more associations can be supported between the pair; and
- Operations to end (normally) and abort (abnormally) an existing association between the initiator and responder interfaces.
- A read-only attribute, **tc_context_setting**, which was populated at the time the responder object was created.

The operation to begin an additional new association was defined to allow (as an implementation option) multiple simultaneous associations between an already created initiator/responder pair. If this feature is supported, then it can significantly reduce the number of objects that need to be created, thus improving scalability. It also reduces the number of method invocations required for association processing by eliminating the need to create responder objects. Recall from the factory interface generated during ST that the creation operation for the responder interface also takes the parameter **a_id**, which creates the first, and perhaps the only, association between the initiator/responder pair. The actual state of the dialog is signaled during operation invocations using the **in** parameter **TcContext** or **TcLinkedContext** (which includes within it a **DialogFlowCtr** parameter) in every ST-generated operation signature.

It is certainly possible to remove this knowledge of the association state, i.e., the TC-specific behavior, from the initiator and responder CORBA objects. This can be done at object creation time by choosing appropriately the value of the attribute **tc_context_setting**. All implementations must be able to support the choice of a value for this parameter that indicates that there will be no support for signaling association state information, nor will multiple associations be supported between an initiator/responder pair. (Note that the support of TC/ROS invocation and linked ids are unavoidable; so it is not possible to completely shield the CORBA programmer from TC/ROS, although default values can be used for these in all operation signatures to at least partially mitigate the effect.) If this value is chosen for the attribute, then it is the function of the gateway to know the exact TC PDU sequencing for a given TC application when interacting with an external TC-User. As this is quite application-specific, it could mean putting a lot of application-specific details at a gateway, which makes it considerably less generic. This information would probably be embedded in the implementation of specific proxy objects at the gateway.

There are also other optional values for the **tc_context_setting** attribute to relax the two interaction semantic constraints imposed by knowledge of TC dialogs.

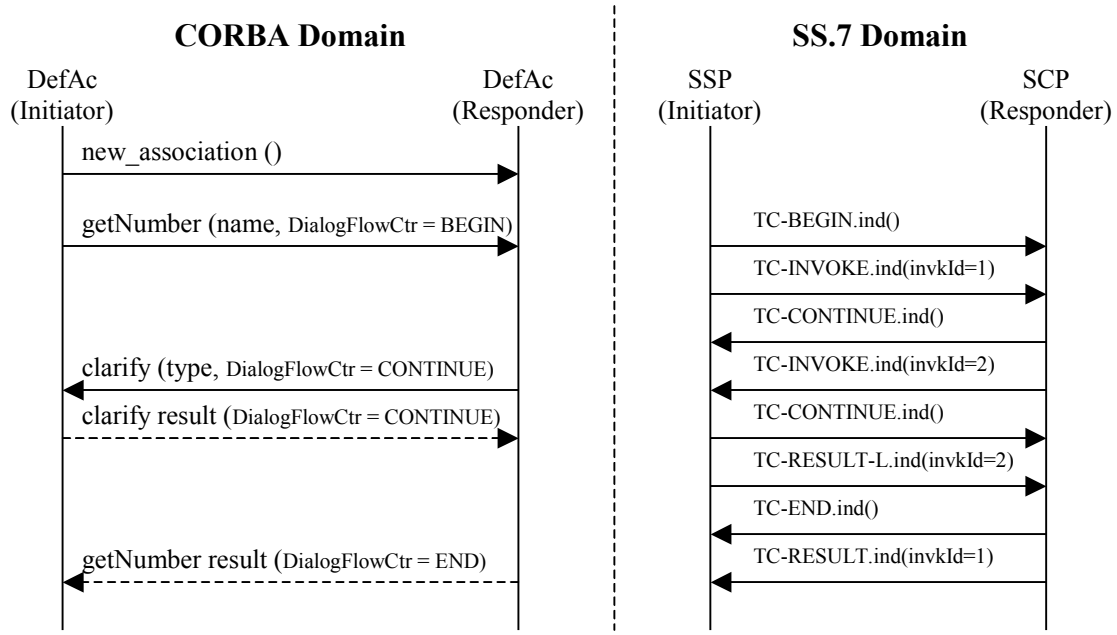


Figure 4.10: TC-User Association/Dialog Maintenance

We illustrate the use of the **DialogFlowCtr** member of the **TcContext** parameter of every TC-User operation introduced during specification translation in figure 4.10. The transfer of this information to the CORBA domain allows us to preserve the TC interaction semantics at both the dialog handling (PDU flow control) and component handing/ROS levels. The availability of this information in the CORBA domain greatly simplifies the construction of TC/CORBA gateways for TC-User object implementations that support this behaviour. The figure uses our example specification translation TC-User definition to show how operations (and parameters) in the CORBA domain retain all the information transferred in the SS.7/TC domain.

4.3.4. Mapping TC/ROS operation/error codes to IDL signatures

TC/ROS operations and errors are given local (integer) or global (ASN.1 Object Identifier) values to distinguish them when encoded in PDUs. CORBA has no concept of such codes. The actual operation or exception name in an IDL signature is encoded in the GIOP message. Thus, one feature of the dynamic translation needed at a gateway is to translate between TC/ROS invocations and CORBA operations. The interface **TcRepository** standardizes access to the information contained in a TC User repository generated during ST (see Table 4.3 for a schematic layout of such a repository for the example ST), and allows the retrieval of an IDL scoped name for an operation or exception (as stored in the Interface Repository (IR)) given its TC/ROS identifier and the interface name of the target object,

and vice versa. This mapping involves two calls: one to the **TcRepository** to get the IDL scoped name for the operation/error as stored in the IR, and a second call to the IR to "lookup" the signature associated with this scoped name.

While this mapping can impose a performance penalty in the case of a fully generic gateway, specific implementation scenarios can offer some panacea. For instance, if the gateway is implemented using CORBA static stubs and skeletons – in other words the gateway has been built for inter-working with a specific TC-User application – then the information in the TC Repository and IR for that application could be implicitly associated with the stubs/skeletons.

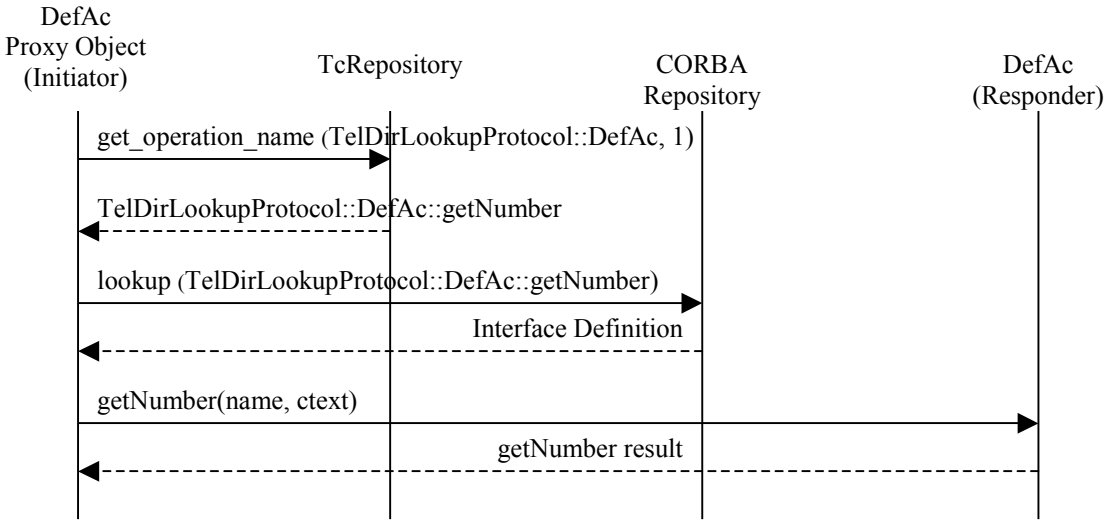


Figure 4.11: A Proxy Object at a Gateway using the TcRepository Interface

In figure 4.11 we see the use of the **TcRepository** interface illustrated using our example specification translation. On the left we see a dynamic implantation of a **DefAc** proxy object at a CORBA-TC gateway. Before the events illustrated in the MSC, a SS.7-based implementation of our TelDirLookupProtocol client has established a dialog with the gateway and sent an operation with an operation code of 1. Our generic TC-User proxy object doesn't statically implement the TelDirLookupProtocol so it must find the CORBA definition of operation in question in order to be able to dynamically marshal it correctly. Hence it queries (using the operation "**get_operation_name**") the **TcRepository** for the IDL scoped name of this operation in the standard CORBA Interface Repository. Once the scoped name is returned it can then find the definition the of the corresponding IDL operation and dynamically marshal an appropriate request and send it to the **DefAc** responder for processing.

4.4. TC PDU-oriented Interfaces

The TC PDU-oriented interfaces are designed to standardize access by TC-aware CORBA objects (such

as proxy objects at a gateway) to a TC/SS.7 protocol stack. This allows implementations of gateways that are independent of a particular stack vendor. At this time, most stack vendors offer proprietary APIs to their TC/SS.7 stack. There is also an effort [JAIN] to provide a Java-based API for this interface. It is not, of course, necessary to use the TC PDU-oriented interfaces to implement a TC/CORBA gateway, as the custom mapping onto a particular TC/SS7 stack may be a part of the implementation of the proxy interface generated during ST. However, these interfaces can be useful if there is a need to build a distributed gateway that is not too closely coupled with the stack platform and hardware. These interfaces represent a very low-level mapping that requires users to be aware of the TC service primitive interface defined in ITU-T Rec. Q.771 [Q771]. It is also unusual in that it requires users to be able to encode/decode ASN.1/BER data based on ITU-T Rec. Q.773 [Q773]. These interfaces could also be used to build TC-aware CORBA applications that do not rely on the ASN.1 to IDL translation algorithms specified but instead use some proprietary mechanism.

The full details of the TC PDU-Oriented interfaces, including IDL, are specified in Appendix 1.

4.4.1. Overview

The interface **TcPduProvider** is an encapsulation of the TC/SS.7 stack service interface/API and is created by a **TcPduProviderFactory**. A TC-aware CORBA object that wishes to communicate with an external system using TC/SS7 messages obtains a reference to the **TcPduProvider** interface by invoking a create operation on the **TcPduProvderFactory** while providing a call-back interface, **TcPduUser**, as an **in** parameter. The factory creates the interface if one does not already exist, and also returns the first dialog id to be used for bi-directional operation invocations between the **TcPduProvider** and the **TcPduUser** interface. If no more dialogs can be supported via that interface, an exception is thrown.

Another operation on the **TcPduProviderFactory** interface allows a TC-aware CORBA object to register to receive any TC PDUs destined for a particular Global Title and, optionally, a specific Application Context. The registering object provides its call-back interface, **TcPduUserFactory**, and the factory throws an exception if more than one object registers to "listen" with a particular GT/AC combination. There was no clean way to allow more than one object to register to listen for TC messages destined for a given GT/AC pair, as it was not possible to identify who the recipient should be for any given received message.

Figure 4.12 illustrates the case when a TC BEGIN APDU is received with the GT/AC combination for which there is a registered listener, first the listener's factory is contacted to get an object reference for a suitable **TcPduUser** (step 4 in figure). Then the contents of the TC-BEGIN PDU are sent to the listener's **TcPduUser** interface using a **begin_ind()** operation, closely resembling in content and semantics the corresponding TC-BEGIN indication service primitive. An **in** parameter in this operation

signature provides the object reference of the **TcPduProvider** interface (which reference is made known to any proxy object created at the gateway) through which all subsequent two-way message exchanges with the external system will take place. After processing the contents of the TC BEGIN PDU, the **TcPduUser** responds with a request to send a TC CONTINUE PDU containing the RESULT of the invoked operation (step 6 in figure). This is then encoded and transmitted across the SS.7 network to the originating TC-User, a SSP in this case (steps 7 and 8 in the figure).

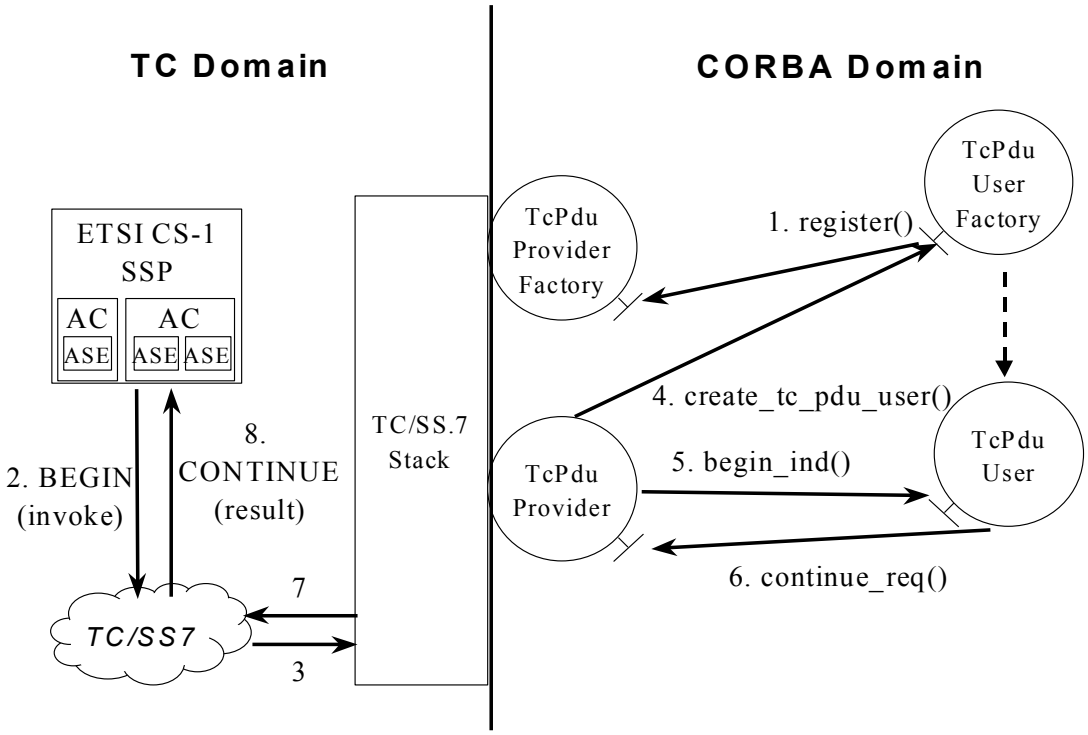


Figure 4.12: TC PDU-Oriented Interfaces upon Initiation of a Dialog from the SS.7 Domain

4.4.2. TC-PDU-Oriented Interfaces and TC-User Objects in a Gateway

The previous section described the capabilities of the TC PDU-Oriented Interfaces in isolation. The primary motivation for creating them is to encapsulate a wide range of vendor-specific SS.7 stack APIs at an inter-working gateway. The question then arises how the TC PDU-Oriented Interfaces should be combined with the generated TC-User and TC-User Factory interfaces to create Proxy objects in a gateway implementation. In general there are many ways of combining the interfaces and it is possible to create Proxy objects that don't use the TC PDU-Oriented Interfaces at all, in effect their implementation would be specialised to work with a particular SS.7 stack vendor's product. However in this section we describe in general terms how the sets of interfaces may collaborate to implement Proxy objects in a gateway.

Figure 4.13 shows a consolidated example where all the interfaces are brought into play to illustrate a

typical scenario including a gateway. Note that the actual interacting objects are purely illustrative and which interfaces are exposed would, of course, be implementation dependent.

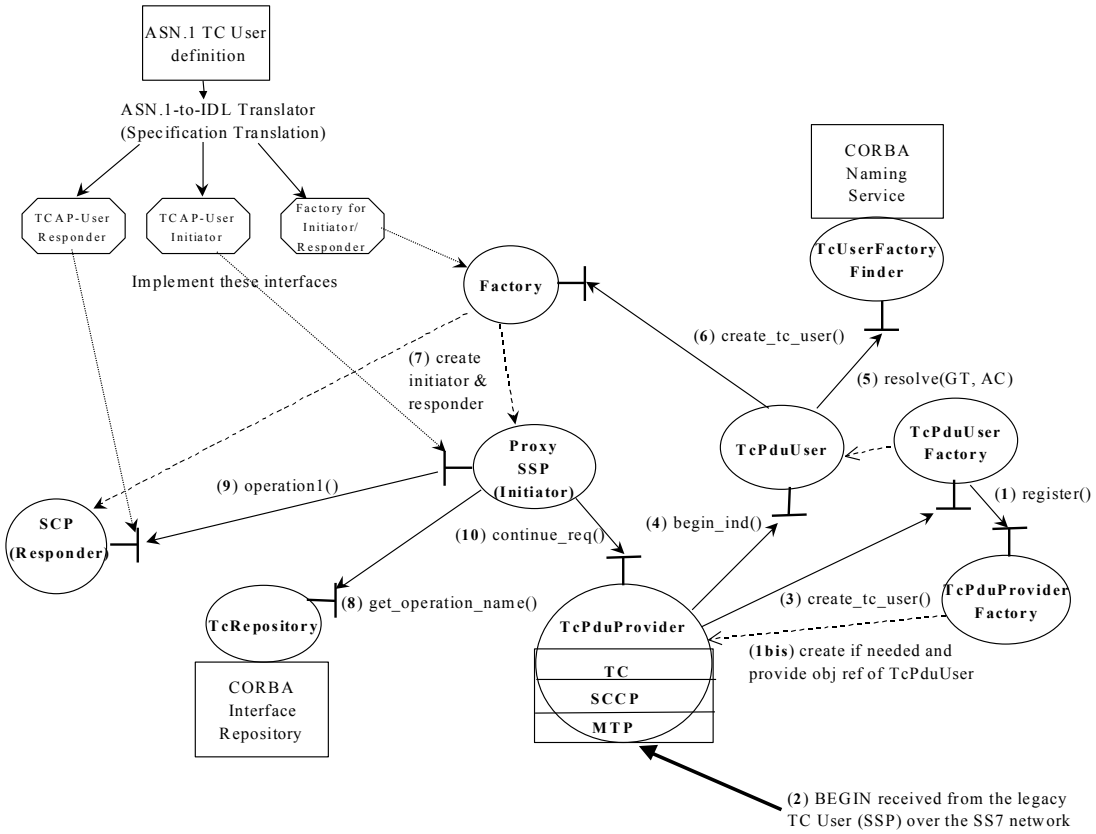


Figure 4.13: An Example of Using the Various Interfaces

The steps are as follows:

1. A **TcPduUserFactory** registers its interest in receiving TC APDUs destined for a particular GT/AC. Its callback interface object reference is passed (step 1bis) to the **TcPduProvider**, which encapsulates the TC/SS7 stack.

2. A BEGIN message with an Invoke component containing an operation1 is received over the SS7 interface from a SSP.

3. The registered **TcPduUserFactory** for the addressed GT/AC is requested to create (i.e. provide an object reference for) an appropriate **TcPduUser** instance.

4. The **TcPduUser** is provided the contents of the BEGIN, in particular the received operation.

5. The received GT is resolved via the **TcUserFactoryFinder** interface (which encapsulates the CORBA Naming Service) to obtain the reference to the Factory object that can create the appropriate initiator and responder interfaces.

6. The **create_tc_user()** operations are invoked on the factory object reference.

7. The factory creates the appropriate initiator (the proxy object for the SSP in the CORBA domain) and the responder (the SCP) interfaces. The creation of the former interface is done first so that the responder interface can be provided its object reference for two-way communication.

8. The ASN.1 operation code for operation1 is converted to the IDL operation name by querying the **TcRepository**.

9. The IDL operation1 is invoked on the responder (SCP) interface.

10. The result of the invocation is returned via the invocation of the **continue_req()** operation on the **TcPduProvider** interface which packages them within a TC CONTINUE PDU which is sent out over the SS7 network to the SSP.

It must be noted that the above represents an abstract architectural model of the events that occur for dialog initiation, realistic implementations may collapse many of the entities described above into coherent sets of executables or processes. However the partitioning described above does provide the suggested elements of standardised distribution of the system on a CORBA platform.

4.5. Deployment Scenarios

In the previous sections we have generally used simplified examples of ASN.1 and generated IDL to illustrate the specification translation process and the workings of the interaction translation features. Realistic ASN.1 and IDL to describe the protocols used by IN systems can easily run to 100 pages of specification. In this section the details of individual interfaces are ignored in favour of examining the structural constraints and degrees of flexibility that the specification allows.

4.5.1. Range of Applicability

The CORBA-based architecture described in the previous sections has great flexibility in terms of implementation strategies and deployment approaches. The Eurescom P508 project [P508] and an OMG white paper [OMGWP] both identified three major applications of CORBA technology to the IN domain. Each of these was considered critical to any eventual deployment of new CORBA-based technology. They are as follows:

- **Migration:** The support of heterogeneous nodes within provider networks consisting of a mixture of CORBA-based and traditional IN nodes. It must be possible to construct adaptation units within the provider network to allow the two types of nodes to communicate.
- **Inter-working:** The support of heterogeneous provider networks where some networks use CORBA technology and others do not. It must be possible to construct inter-working units between

the provider networks to enable communication.

- **SS.7 as a Kernel Transport Network:** This requires SS.7 to be used as an ESIOP within a CORBA domain to enable islands of CORBA entities to communicate over the SS.7 protocol stack.

The CORBA-based architecture presented here addresses the first two concerns. The third concern is an orthogonal issue and addressed by other work [OMGIN].

In addition to the requirements identified above, this work enables all TC or ROS –User protocols to be mapped to the CORBA domain. This includes X.500 Directory Services [X500] and GSM Mobile Application Protocol [MAP]. Both of these specifications are also used in some telecommunications service applications and the broadening of the specification's scope to include them naturally increases the potential for CORBA-based hybrid service platforms.

4.5.2. ETSI CS-1 Example

To examine the interfaces produced by specification translation we will use the ETSI ASN.1 specification of IN Capability Set 1 [ETSI]. This document defines the INAP protocol for three IN functional entities, the SSP, SCP and IP. Each functional entity is further divided into a number of Application Contexts (each governing a specific set of related IN dialogs). Specification translation of this document results in one responder and one initiator interface for each Application Context. The full list of interfaces, broken down by functional entity is as follows. The number of operations included in each interface is also specified as way of estimating relative complexity [LOR].

For the SCP:

- SCP_to_SSP_traffic_management_AC_initiator – 5 operations
- SCP_to_SSP_service_management_AC_initiator – 6 operations
- SCP_to_SSP_AC_initiator – 10 operations
- SSP_to_SCP_service_management_AC_responder – 6 operations
- SSP_to_SCP_AC_responder – 16 operations
- assist_handoff_SSP_to_SCP_AC_responder – 8 operations
- IP_to_SCP_AC_responder – 7 operations

For the SSP:

- SCP_to_SSP_traffic_management_AC_responder – 6 operations
- SCP_to_SSP_service_management_AC_responder – 6 operations
- SSP_to_SCP_service_management_AC_initiator – 6 operations

- SCP_to_SSP_AC_responder – 23 operations
- SSP_to_SCP_AC_initiator – 20 operations
- assist_handoff_SSP_to_SCP_AC_initiator – 15 operations

For the IP:

- IP_to_SCP_AC_initiator – 10 operations

For creating all the above interfaces:

- AeFactory – 16 operations

For a CORBA-based IN supporting ETSI CS-1 these interfaces and the supporting interfaces specified earlier are our basic building blocks. Of course there will also be replication of many of these interfaces in any deployment. We can define object distribution strategies which take advantage of the location-transparency that a CORBA platform provides to enhance the system in terms of non-functional requirements such as performance or fault tolerance [MCA].

Additionally we could utilise the flexibility of the CORBA platform to integrate some of the application contexts above on the same computational node to produce a hybrid SSP/SCP. Even Application Contexts from multiple TC-User protocol specifications could be combined to produce hybrid INAP/MAP functional entities that could provide both mobile and fixed-line services. It is also natural to envisage that these CORBA-based service nodes would incorporate CORBA-based management features as defined by the OMG or 3GPP [MGT].

The remainder of this work only considers the case where the generated interfaces are traditionally grouped to provide CORBA-based IN systems that still have identifiable SSP and SCP nodes. For a consideration of novel distribution polices and strategies see [MCA] and related work.

4.6. Implementation Experiences

In the course of defining the interaction and specification translation solutions presented here a series of prototyping experiments were undertaken. The form and capabilities of two of the prototypes developed are briefly described here. This work demonstrated weaknesses in the initial specification lead to refinements and ultimately motivated the detailed work on Internet Protocol-compatible network transports to support CORBA-based IN platforms presented in the next two chapters.

4.6.1. Initial Prototype – TCAP to CORBA Gateway

This early prototype was based on an SDL implementation of TCAP, which was supported by ASN.1 encoder/decoder libraries generated by the SNACC compiler [SNACC]. This formed the “SS.7” side of the gateway and was connected to a basic CORBA-based IN system capable of performing a number

translation service.

Issues identified with this prototype were:

- TCAP is not the ideal level for inter-working due to both the lack of any obvious structure for the CORBA objects generated by specification translation and the need to have ASN.1/BER encoder/decoders in the CORBA domain. This is why the final specification inter-works at the level of TC-Users. The TC PDU Oriented interfaces may be seen as a successor to this prototype.
- Implementation of gateways is complicated by the need to co-ordinate between multiple closed control loops. Both the ORB used (Orbix) and the SDL execution environment (SDT) needed a closed control loop to check for messages from their side of the gateway. This was worked around by creating a gateway consisting of two communicating unix processes. However the performance overhead for this style of communication was very large and alternate strategies would be used with future prototypes.

4.6.2. Second Prototype – INAP to CORBA Gateway

This prototype was an attempt to build a commercial-quality gateway solution in partnership with industry. The first TC-User protocol supported was IS-136.1 MAP [IS136], an American mobile services standard. The implementation was based on the Ulticom OMNI SS.7 protocol stack [OMNI] and the Visibroker CORBA ORB [VISI]. A detailed set of requirements and design specification documents were developed which illustrate the application of our IN architecture to an actual implementation.

Figure 4.14 shows the architecture of the gateway at the sub-system level. A brief description of the gateway is as follows:

- Each gateway "network element" consists of a Proxy Service, a SS.7 Service and a Gateway Management Service sub-system. Due to the CORBA implementation all of these sub-systems may be distributed or replicated as desired to achieve the necessary levels of performance, fault tolerance, etc.
- There is one Proxy Service for each TC-User protocol that the gateway can translate, for example ETSI CS-1 INAP. This provides the specification translated proxy and factory objects and the TcPduUser interfaces.
- There is one SS.7 Service for each physical SS.7 signalling point in the gateway. This encapsulates and makes location transparent the local SS.7 API through the TcPduProvider interfaces.
- The local Gateway Management System provides various support functions for the node such as alarm generation, configuration management, etc. It essentially acts as a CMIP Management Agent.

- There is a remote management subsystem, which may manage a number of gateway "network elements". This also connects to a standard CORBA event network and provides CORBA-based TMN management events to a network management centre.

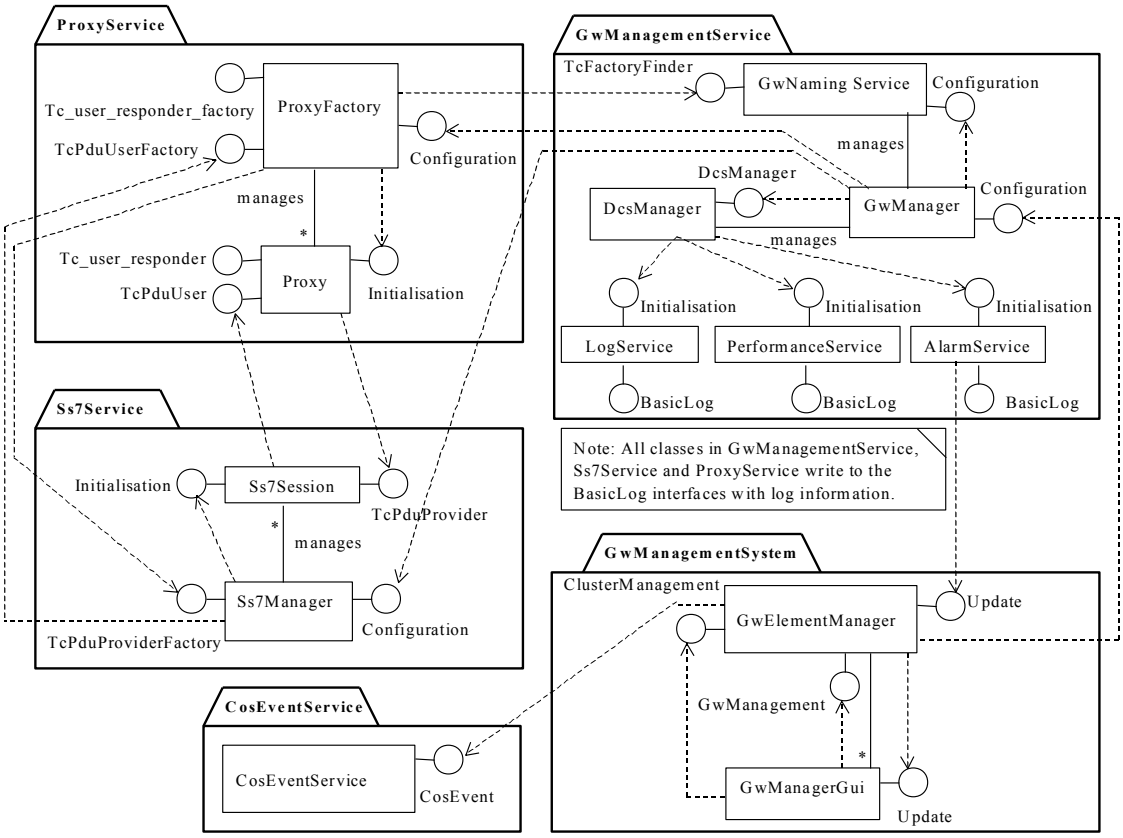


Figure 4.14: Subsystem Structure of the Second Prototype (UML Class Diagram)

This prototype was close to production code and a significant effort was made to integrate it with a commercial mobile service centre (MSC). Many of the interaction translation interfaces were refined in the course of this work. However the key finding was that a CORBA platform running over the IIOP protocol (and hence TCP/IP) was unsuitable for telecommunications signalling applications. This was due to the high availability requirements placed on such a platform. These conditions could not be met while using TCP as the transport protocol. This was due to the best effort nature of the traffic and insufficient ability to configure protocol parameters on a per-session or per-application rather than per node basis.

4.7. Comparison with Earlier Work on CORBA-IN

The previous sections have described our architecture for developing CORBA-based IN application entities and gateways. This work was used as the basis for the OMG specification for IN-CORBA inter-

working. In this section we compare our work with the earlier work of Mazumumdar and Mitra [MAZ, MAZ2] on CORBA-based IN solutions. In many ways our work is a direct continuation of their work and several of their key approaches are retained. The most significant areas of new work here consist of:

- Formal definition of specification translation algorithms for all forms of ASN.1 specification.
- Explicit modelling of TC dialog semantics in the CORBA domain.
- A refined model of IN-CORBA gateways that abstracts away from implementation details.
- Many minor changes and improvements designed to maximise performance.

Each of these is briefly contrasted with the earlier in the following subsections.

4.7.1. Formal Specification Translation Algorithms

In this area of the architecture many ideas of Mazumumdar and Mitra have been developed to produce a complete ST algorithm that is capable of being implemented as a compiler. Their work illustrated some ST concepts through a worked example that was manually translated. However there were many important cases not considered that a mapping had to be defined for. This included the development of specification translation mappings for TC-User specifications written in pre-1994 ASN.1 (used by most INAP specifications) that utilise the old ASN.1 MACRO notation rather than ROS Information object constructs. Our work identified two important sub-cases of pre-1994 ASN.1 specifications: ones like ITU-T CS-1 INAP which rely on natural language to define groupings of operations into contracts or application contexts and MACRO based Application Service Element/Application Context specifications. Another omission of their work was a lack of any definition for the mapping of ASN.1 Extension information objects or MACROs. This important construct is often used in the definition of INAP (or other TC-User) operations and errors.

One aspect of the earlier ST work that we heavily revised was the dependency on TINA-C extended IDL (object description language, ODL). This means that standard OMG IDL can be used to model TC-User entities in the CORBA domain. Since there are no commercial ODL products available and the TINA-C initiative has stalled this makes the specification much more useful for actual deployment. Our model of TC-User entities in the CORBA domain was brought closer to the OMG object management architecture through the definition of TC-User factory objects generated during specification translation. The existence of these manager objects increase deployment and implementation flexibility by allowing load balancing or other object distribution strategies to be adopted. It also means the architecture now follows best OO design practice as specified by standard OO design patterns [GHJV94]. Our work defines both generic factories for abstract TC-User objects for use with the dynamic CORBA programming model (e.g. at gateways) and specialised factories for each type of

generated TC-User object for statically bound implementations (e.g. in native CORBA implementations).

Finally we formalised the storage of important TC information that was lost during Mazumumdar and Mitra's specification translation. This included the values of operation timers and operation and error codes. This information is never used in the CORBA domain but its availability is essential for building a viable gateway. Our interaction translation solution includes the specification of an interface, the **TcRepository**, which standardises access to this data in the CORBA domain.

4.7.2. Modelling of TC Dialog Semantics

The earlier work on IN (TC-User) protocol modelling in the CORBA domain focused on ROS constructs. This approach was attractive because it addressed the common aspects of the SS.7 and OSI protocol stacks. However, in the signalling domain the TC layer running over SS.7 contains both component handling (identical to OSI ROS) and dialog handling sub-layers. Thus CORBA objects built under this scheme were missing half of the information required to inter-work at the TC level. It would be possible to build a relatively heavyweight gateway solution which included information on the dialog handling specifics for a subset of all IN protocols but truly generic gateways could not be built. Additionally we found that there was a performance advantage to minimising the number of CORBA objects that had to be instantiated to handle a given volume of IN service requests. Explicitly modelling the TC concept of a dialog as an association between two CORBA objects gave us the ability to solve both of these problems. (The latter problem was solved as a result of allowing multiple associations to exist between a given pair of objects.)

This naturally defined a set of interaction semantics that all CORBA-based TC-User objects must follow. We were able to support these special semantics with the definition of base classes for TC-User objects and TC-User object factories that included dialog specific operations. In addition, the earlier use of an operation context parameter (inserted into the signature during specification translation) was extended to include dialog-handling information. The use of dialog context information was extended to operation returns and exceptions as is normal in the TC domain. The earlier ST had omitted this information. Our context information also provided a natural location for the support of the TC concept of linked operations. In addition we included support for the 1993 ROS/TC ability to pass dialog specific data at bind time.

In order to not place to heavy a burden of TC knowledge on CORBA implementers the support of these dialog features was made optional and an attribute defined to allow peer objects to query the level of TC dialog support available from a given TC-User object.

With this new model of TC-User object interaction semantics in mind we were able to remove several

cumbersome features of the previous work in this area. This included the ill-defined "contract service" which was replaced with access to the standard CORBA naming service via our **TcFactoryFinder** interface and syntactic constraints on object creation that enforced TC interaction semantics.

4.7.3. Refined Model of TC-CORBA Gateways

One completely new area of specification in our work was the definition of a standardised API for access to TC/SS.7 protocol stacks. Previous work had not addressed this issue at all and as a result dwelt on many implementation details of the gateway. Our API model influenced the work of the JAIN standardisation groups in this area, which were just beginning to address this issue when our work was published.

Our work on elucidating the use of proxy objects at gateways and the strictly enforced location transparency in our interaction translation means that it is now feasible to discuss gateway issues at a much higher level of abstraction that was previously possible. This also opens up many more degrees of freedom in the design and implementation of such gateways. As stated above, one consequence of modelling all TC aspects (i.e. dialogs as well as operations) in the CORBA domain is that truly generic TC-CORBA gateways can now be built using the CORBA dynamic programming interface instead of only allowing for protocol-specific gateways. This is an important property of a gateway solution because there is a huge number of INAP and related protocol variants available in the IN marketplace due to vendors' proprietary extensions to the protocol.

Finally, as part of our interaction translation specification we defined a number of new helper interfaces for gateway implementations. These standardise access to interaction translation features such as Global Title translation and gateway-specific features such as ASN.1 operation code mappings to IDL operation signature definitions. As these functions are necessary for the operation of a gateway this makes our solution more comprehensive.

4.7.4. Performance-based Improvements

A key non-functional requirement for signalling systems is the ability to meet standardised performance (processing delay, mean handling time, etc.) metrics. Through careful design of IDL interfaces it is possible to minimise message encoding processing times and enable the deployment of scalable systems. There are several examples in our final architecture where such performance considerations have influenced the design. This is in contrast to the earlier work on IN-CORBA, which ignored such issues.

The most important manifestation of our performance-oriented approach is in the ability of our TC-User objects to support multiple, simultaneous associations (dialogs). This ensures that object creation overheads are not automatically built into each service request. In fact the need for explicit association

start messages are also eliminated via the use of implicit association creation through the use of a new association ID coupled with a context parameter of "BEGIN". This saves one remote method call per service request. However the flexibility of our approach is such that dynamic object creation is still supported if a given implementer wished to pursue that route.

With our IDL it is possible to see many places where previously generic constructs such as name-value pairs were used, for example to carry dialog context information. In all possible cases we have replaced these with statically declared IDL constructs that explicitly model the most frequently used types and values, such as the operation code (ID) for a given operation. This leads to significant speed-ups in parameter processing and hence service execution. This is due to the relatively cumbersome processing necessary to deal with dynamic types such as name-value pairs. Once again we retain the availability of name-value pairs for truly unusual context information so no loss of generality is envisaged by our approach.

4.8. Summary and Conclusions

This chapter has described a new architectural framework to support the implementation of CORBA-based distributed Intelligent Network (TC-User) systems. This framework consists of the following elements:

- An algorithm for mapping TC-User protocol ASN.1 specifications into IDL specifications (specification translation).
- A group of standard services with well-defined IDL interfaces to support IN systems and gateways implemented as CORBA systems.
- A set of robust interaction semantics for TC-Users in the CORBA domain that provide location transparency for TC-User entities that interact across heterogeneous implementation domains (interaction translation).
- A standard set of interfaces for access to SS.7 protocol stacks in the CORBA domain.

This framework was mainly discussed through reference to an example system mapping. A complete specification of the mapping algorithm can be found in Appendix 1. We also discussed deployment scenarios, prototyping experiences and the relative benefits of our proposal compared with earlier work on CORBA-IN integration.

The framework provides new standardised degrees of freedom for IN system implementers to leverage distributed object technology. It is anticipated that the benefits of this include:

- Development of cheaper service platforms based on off the shelf distributed processing environment solutions (realtime ORBs).

- Ease of integration of multiple service (IN, MAP) and management protocols (CORBA-TMN) through use of a common data format and remote procedure call mechanism (the ORB).
- Enhanced possibilities for distributed load control, scalability and fault tolerance due to the specification of smaller standardised elements of service control functionality (as distributed objects).
- Through abstraction it eases the transition from expensive SS.7 networks to standard networking technologies such as the TCP/IP family of protocols.
- Integration with other technologies and standard services within the CORBA framework to enable development of novel services that draw on elements of both telecommunications and information technology.

The framework additionally provides comprehensive support for interworking between CORBA-based and traditional IN nodes. This is seen as vital to the deployment of any new solution for network services as previous attempts at new service architectures have failed by ignoring this aspect. The work presented here is general enough to deal with any TC-User protocol such as INAP, MAP or DAP and as such addresses a much wider set of solutions than simply CORBA-based IN.

The work presented here has been tested through the development of working prototypes and passed through the formal reviews of the standardisation process of the Object Management Group (OMG). In 2001 it was adopted as an official specification of the Telecommunications Task Force within the OMG.

The standardisation and prototyping effort carried out in support of this specification also pointed out several issues that require further investigation for any realistic chance of deployment of the framework. These include:

- The lack of availability of commercial ORB products with suitable realtime capabilities to act as telecommunications service platforms. However proprietary ORB implementations with these capabilities do exist.
- Concerns about the likely performance impact of the lengthy IDL generated by specification translation.
- A lack of understanding of the issues involved when attempting to use the inter-ORB protocol, GIOP, and TCP/IP as a transport mechanism for realtime telecommunications signalling data.

As the first two issues are largely dependent on ORB implementation specifics and the likely candidate ORBs are still closely guarded commercial secrets it was decided to concentrate on the final issue for the rest of this work.

Chapter 5. SCTP, The DIN Network Transport

In this chapter we examine in detail Stream Control Transmission Protocol (SCTP), the most likely protocol to provide network transport services for the signalling generated by the DIN architecture. We develop a useful analytic model of multi-homed SCTP endpoints, describe our event-based simulation model of SCTP, validate our analytic model through simulation and experiment and finally investigate the robustness of SCTP congestion control mechanisms.

5.1. Introduction

In this section we first discuss the likely candidates for a DIN network transport protocol and then we provide an overview of our SCTP modelling approach.

In the following sections we present our models for two SCTP endpoints communicating and the results of a series of experiments performed to verify these models and investigate SCTP behaviour. This work detected several flaws in the first specification of SCTP and produced a useful first order approximation of SCTP multi-homed behaviour for network planning purposes.

5.1.1. Potential Candidates for DIN Network Transport

In chapter 2 we discussed the networking architecture for CORBA environments. Figure 2.4 illustrates the three likely standard transport protocols for DIN applications. These are the General Inter-ORB Protocol (GIOP) over TCP/IP (usually called the Internet Inter-ORB Protocol (IIOP)), GIOP over SCTP/IP or GIOP over SS.7's SCCP transport (SCCP-IOP). Another possibility for telecommunications applications are propriety IOPs developed by individual equipment but they are not considered further here due to the lack of publicly available information about them. We now briefly review the possible transport protocols/IOPs for DIN applications.

5.1.1.1. GIOP over SS.7 (SCCP-IOP)

This protocol was developed as part of the IN/CORBA Inter-working architecture [OMGIN] to provide both an alternative means for ORBs to be integrated with the existing IN architecture and as a complimentary specification to the DIN specification. If not used in conjunction with the DIN specification it allows the tunnelling of inter-ORB communications over SS.7 networks, when used with the DIN specification it allows transport of DIN messages in a SS.7 environment. It is the latter use that we consider here.

It is unlikely that any DIN system will use this IOP as to the author's knowledge there are no commercially available implementations of this specification and a major driver for DIN adoption is to reduce the great costs associated with the current IN/SS.7 infrastructure. Keeping SS.7 as part of the solution would allow for evolution to new systems but is unlikely to bring the short-term cost savings that telecom operators desire. Re-using the existing SS.7 network would also retain the current partitioning of the telecommunications network into separate signalling and bearer networks. Another key driver for current telecommunications research is the goal of a unified network based on IP that will only be virtually partitioned through the use of DiffServ [Car98] or other IP QoS technologies. It is anticipated that this unified telecommunications network based on IP will provide great cost savings and new degrees of flexibility as a common infrastructure transports, signalling data, bearer voice communications and multi-media communications. For all of these reasons GIOP over SS.7 is not considered further in this thesis.

5.1.1.2. GIOP over TCP/IP (IIOP)

This is the baseline network transport protocol for the CORBA DCE [CORBA]. It is well established and many commercial implementations are available. Initial prototyping for DIN environments (see §4.6) made use of this IOP. However there are serious concerns within the telecommunications industry about the ability of TCP to perform reliably as a signalling transport [Coe02, Coe03]. These concerns can broadly be categorised as follows:

- issues with responsiveness to failure - TCP is a best-effort protocol making it generally unsuitable for time sensitive data transport.
- redundancy - SS.7 has comprehensive protocol support for engineering fault-tolerant networks and standard TCP has none.
- behaviour under network congestion - TCP's congestion controls operate in a feedback loop that invalidates the assumptions of standard teletraffic techniques such as the Erlang B formulae.

It is important to note that this criticism is largely directed at TCP whereas the IP network protocol, especially with future extensions, is generally seen as suitable for a wide range of telecommunications applications. For these reasons GIOP over TCP/IP has not been considered a realistic IOP to be used for DIN applications and is considered no further here.

5.1.1.3. GIOP over SCTP/IP

Within the IETF there has been a recent standardisation effort directed at solving the problems of TCP for telecommunications signalling applications. This work has resulted in the Stream Control Transmission Protocol (SCTP) [SX00] and a family of adapter-layer protocols that can mimic the services of individual SS.7 layers such as MTP2 or SCCP. This allows traditional telecommunications

signalling applications to run over an IP infrastructure and is the cornerstone of the International Packet Communications (formally Softswitch) Consortium's [IPCC] approach to building next-generation telecommunications network elements and service applications.

Despite it's signalling transport origin, SCTP is also a general-purpose transport protocol and shares many characteristics with TCP. In particular, the 3GPP has deemed that it is the correct transport for signalling traffic between CORBA-based applications within their service architecture [3GSCTP]. Work is ongoing in the OMG to standardise a SCTP mapping for GIOP [OMGSCTP]. Given its suitability for telecommunications signalling applications and the availability of an IOP based upon it, SCTP seems to be the most suitable standards-based candidate for DIN network transport.

SCTP is a relatively new protocol and although it's behaviour is based on TCP with extensions for telecommunications environments there has been relatively little published work investigating it. This is partially due to the fact that most of the SCTP development work up to this point has been led by industrial R&D rather than basic research. This has led to less information being published on prototyping experiments and theoretical evaluations. It is also due to the fact that SCTP represents an agreed compromise by industrial partners with many years experience of implementing proprietary or ad-hoc solutions for transporting signalling traffic over IP networks. Most of these solutions were based on UDP rather than TCP and involved the development of an additional layer on top of UDP providing the required transport services, such as fault tolerance. This means that considerable practical experience was brought to bear on the development of SCTP. However the final SCTP protocol standard included many features, such as TCP-like congestion control, that were not originally envisaged in the design. Some of these features were a result of IESG input into the standardisation process and some were based on current TCP research. Important influences on the final specification was the new requirement for the transport to be applicable to "big I" Internet usage in addition to the well-engineered closed IP-based core signalling networks that most telecommunications providers operate.

As a necessary step towards the goal of a DIN platform based on CORBA and SCTP, our work has focused on addressing these gaps in the published information on SCTP. Hence we study SCTP congestion control behaviour in detail, develop models of SCTP nodes and report on experiments carried out both in simulation environments and lab-based test-beds.

5.1.2. Overview of SCTP Models

As discussed in Chapter 3 we can broadly divide modelling activities into event-based simulation and analytic modelling. Traditionally, simulation models are useful for verifying network operation in limited scenarios but a broader, analytic characterisation of SCTP is required to perform vital

telecommunications network planning tasks such as dimensioning in a reasonable timeframe. A number of studies [PF97] have shown the inapplicability of traditional teletraffic techniques to TCP networks. SCTP exhibits the same sort of dynamic response to network congestion as TCP and so is equally unsuited to these traditional techniques. This increases the relative importance of simulation and experiment compared to analytic modelling for TCP or SCTP. Another consequence of the general inapplicability of analytic techniques such as queuing theory is that when developing analytic models of SCTP networks we are forced to go back to more fundamental probabilistic modelling techniques.

5.1.2.1. Analytic Model

Recently there have been a variety of publications that provide analytic characterisations of the behaviour of TCP sources under limited conditions, for example [Pad00, MSMO97, OKM96]. Typically these deal with the steady-state behaviour of persistent or greedy traffic sources with an assumption that packet losses are independent. Indeed the accurate characterisation of TCP traffic sources forms an entire field of research at the present time, for example [Vei01, OB01, McC00]. The information provided by the TCP models is not generally sufficient to engage in performance modelling of a network in terms of transient behaviour but a number of useful network characteristics, for example average throughput, may be predicted. Any deployment of SCTP as a DIN transport will require such tools to support network planning. Thus a characterisation of the behaviour of steady-state SCTP sources is a priority. In general the TCP models will suffice for single-homed SCTP sources but given the multi-homed, fault-tolerant network environment demanded by signalling applications it is necessary to develop suitable multi-homed models for SCTP.

Our approach has been to characterise a multi-homed SCTP endpoint as a combination of two related sources, one a persistent source operating in congestion avoidance mode transmitting on the primary route and the other a transient source operating in slow start mode transmitting on the secondary route. As the amount of data to be transmitted on the secondary route has a simple relation to the loss rate on the primary route, given the loss rate on the primary we can estimate the amount of data to be transmitted on the secondary route. This allows us to re-use models developed for TCP connections in congestion avoidance and slow-start and combine their results to predict the behaviour of multi-homed SCTP source with a primary and a secondary route.

5.1.2.2. Simulation Model

The basis of our simulation model is a complete implementation of SCTP in the OpNet modelling environment. This includes all features, states and packet formats of the SCTP protocol as specified by the IETF. We also developed a basic SCTP-user application model that can initiate and accept multi-homed SCTP associations. In addition the application model can generate a wide range of different

demands on the SCTP transport (variable association start time, variable application data size, variable destination, variable offered data rate).

A single multi-homed SCTP node can then be modelled as a number of instances of our application model and a single instance of our SCTP model. The OpNet environment then provides standard models for the IP protocol, IP router nodes and physical links. In the modelling tool we were then able to configure arbitrary network topologies of multi-homed SCTP/IP nodes and routers. Experiments were then performed by establishing a given network configurations and simulating the behaviour of these SCTP networks whilst collecting data on network performance.

5.2. Analytic Model of Multi-homed SCTP Flows

In an effort to establish what are the likely performance characteristics of DIN platforms using SCTP as a network transport we must first establish the performance of the underlying SCTP network. The most likely deployment of SCTP as a signalling transport is as multi-homed endpoints, which have multiple network paths between nodes for redundancy purposes. This is similar to the topologies adopted by network operators for the SS.7 network. Our approach is to re-use and combine suitable TCP models for the likely behaviours on the SCTP primary and secondary paths. Our aim is arrive at a computationally efficient expression so that when it is generalised to treat networks of multi-homed SCTP flows in Chapter 6 it will result in acceptable calculation time for an iterative network analysis method.

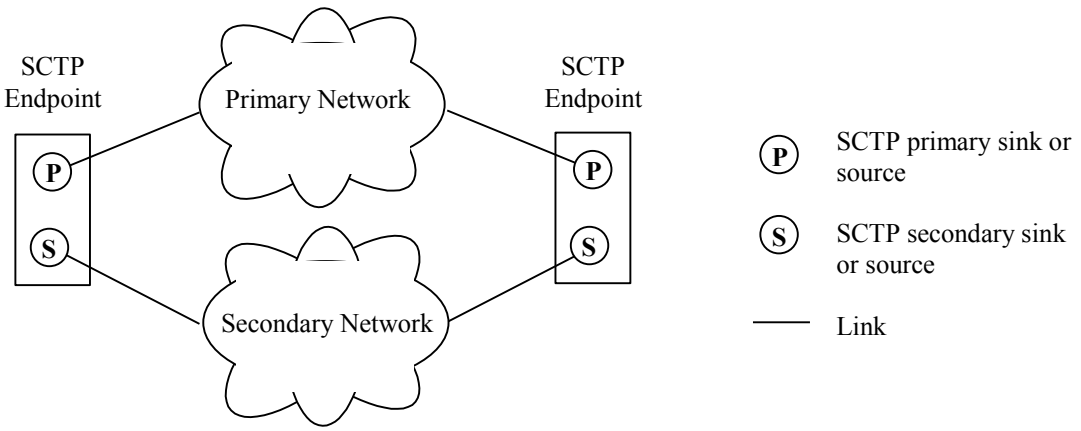


Figure 5.1: Characterising a SCTP Multi-homed Endpoint

Our basic modelling assumption is that a SCTP multi-homed endpoint can be treated as a group of coupled TCP sources. See figure 5.1 for a schematic diagram of the combined system. The primary path (across the primary network) is assumed to behave similarly to a persistent TCP source that experiences independent packet losses i.e. it operates in congestion avoidance mode. The use of a persistent source

for the primary path is of course a simplifying modelling assumption. However given the ability of SCTP applications to use multiple streams within an association it is likely that a transaction-based system, such as telecommunications service signalling, will take advantage of this ability to maintain a common congestion window over a wide range of logical connections (realised as SCTP streams) [Coe01]. Thus a connection between two signalling nodes is more likely to behave as a persistent source from the point of view of SCTP congestion control. Any additional paths (secondary paths) are assumed to behave similarly to TCP sources with limited amounts of data to send i.e. they operate in slow-start mode. This is because in normal operation SCTP only uses the secondary path for redundancy (sending all retransmitted data on that path) rather than load balancing as is used in the SS.7 environment. Under normal condition this corresponds to having small amounts of data to send and hence the connection will never leave SCTP slow-start mode. The two networks are assumed to be independently provisioned, which simplifies the analysis of the combined system. First we describe the relationships between throughput and packet loss probability for SCTP sources in congestion avoidance and slow-start modes and then develop the relationship between SCTP primary and secondary sources in a multi-homed SCTP endpoint.

5.2.1. Modelling a SCTP Route in Congestion Avoidance (Primary Route)

Recently a number of models [Pad00, MSMO97, OKM96] have been developed that characterise the TCP steady-state throughput for persistent sources as a function of operating conditions (especially observed packet loss rate). This corresponds to the behaviour of a TCP source in congestion avoidance mode. These analytic models are based on a number of modelling assumptions such as random independent packet losses but have been shown to predict TCP behaviour for a wide range of real-world operating conditions [Pad00, MSMO97]. Re-using this approach is valid for single-homed SCTP sources or, as here, the primary route of a SCTP multi-homed association as any minor differences in protocol behaviour introduce smaller differences in predicted results than the errors in the model's predictions. [BR02] (also see § 5.5). In fact SCTP satisfies at least some of the modelling assumptions better than traditional TCP variants such as Reno TCP because SCTP implements SACK-based flow control which minimises the likelihood of isolated packet losses leading to timeout rather than fast recovery behaviour [MSMO97].

The primary relationship describing the behaviour of TCP connections is between the mean congestion window size ($\overline{w}$) of a persistent TCP source (i.e. with infinite data to send and operating in congestion avoidance mode) and the packet loss probability (p) on the route it traverses. In general these relationships are of the form:

$$\overline{w} = \frac{k}{\sqrt{p}} \qquad (5.1)$$

Where the value and complexity of the expression *k* depends on the details of the model formulation and the aspects of TCP window behaviour being described by the model. In the work that follows we use the simplest expression commonly used where $k = 1/\sqrt{2}$. More complex expressions could be used but they increase computational complexity for relatively small returns in accuracy [Pad00]. The method described here could be used with the more complex expressions for k but for ease of explanation we use the simpler approximation. Thus (5.1) becomes:

$$\overline{w} = \sqrt{\frac{2}{p}} \qquad (5.2)$$

Where $\overline{w}$ is the mean congestion window size of a persistent TCP source and *p* is the packet loss probability on the route it traverses. As the congestion window size is an absolute limit on the throughput of a TCP/SCTP source and given the value for this we can easily calculate the average throughput of a TCP/SCTP source. The relationship is as follows:

$$\overline{B} = \frac{\overline{w}m}{\bar{t}} \qquad (5.3)$$

Where $\overline{B}$ is the average throughput on the route, *m* is the route message segment size (MSS) and $\bar{t}$ is the average round-trip time (RTT) for the route. Some TCP models [Pad00] include a constraint on the maximum congestion window size (and hence throughput) based on the advertised receiver window (*arwnd*) of the receiver. This constraint is the basis of the well-known Bandwidth-Delay Product rule for TCP connections [St94]. SCTP allows a much larger advertised receiver window than TCP (32 bit instead of 16 bit unsigned integer *a_rwnd* field) and thus unless deliberately configured to restrict throughput it is unlikely that telecommunications equipment will suffer from this restriction when using SCTP. If an advertised receiver window (w_{a_rwnd}) less than the bandwidth-delay product of the link is configured then (5.3) becomes:

$$\overline{B} = \frac{\mathrm{Min}(\overline{w}, w_{a\_rwnd})m}{\bar{t}} \qquad (5.3a)$$

Another constraint on throughput, which these expressions don't take into account, is the maximum bandwidth available on a given route. This will always limit throughput, no matter how low the observed packet loss rate is. If the maximum bandwidth available is denoted B_{max}, then (5.3a) becomes:

$$\overline{B} = \mathrm{Min}\left(\frac{\mathrm{Min}(\overline{w}, w_{a\_rwnd})m}{\bar{t}}, B_{\max}\right) \qquad (5.3b)$$

Another useful property of the route that can now easily be calculated is the average delay experienced

by packets traversing the route. Given some distribution of packet sizes with an average size, $\bar{s}$, the average delay or latency ($\bar{l}$) experienced by packets traversing the route will be given by:

$$\bar{l} = \frac{\bar{s}}{\bar{B}} \tag{5.4}$$

Thus we have a fundamental property of the route, the relationship between congestion window size and observed packet loss rate, and two derived properties, the average throughput and latency on the route. This gives us a useful characterisation of the SCTP primary route behaviour in the presence of a given observed packet loss rate (p).

5.2.2. Modelling a SCTP Route in Slow-Start (Secondary Route)

Our modelling assumptions when partitioning our reference network (c.f. fig. 5.1) result in a secondary network which is lightly loaded and secondary sources that emit limited amounts of data. This corresponds to the secondary SCTP source behaving like a TCP source in slow-start mode. As with the primary route, we wish to characterise the behaviour of this route in terms of the average window size (and hence average throughput and latency) for a given packet loss probability.

For this network we utilise the approach defined in [Eri00]. Thus the evolution of the SCTP window size on this route is characterised as a Markov chain (fig.5.2) where the probability of doubling the window size is:

$$s^{2^j} \tag{5.5}$$

Where s is the probability that a segment is sent and that there is no more data to send on the current connection and 2^j is the number of segments in the current window size.

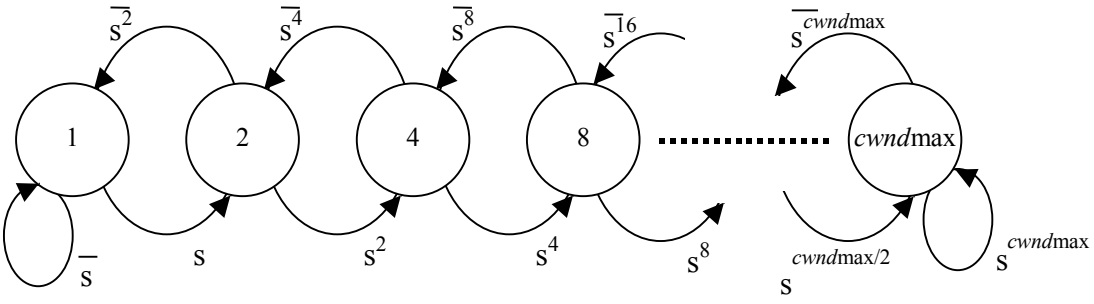


Figure 5.2: Markov Chain Approximation to SCTP Slow-Start Window Dynamics

We assume that the total amount of data to be sent over the route is geometrically distributed. Thus, given a probability (q) that there is more data to send, the average total amount of data to be sent ($\bar{d}$) is

related to q by:

$$\bar{d} = \frac{1}{1-q} \qquad (5.6)$$

Assuming independent losses with packet loss probability p, the probability that there are both more data packets to send and no packet losses (s) is simply:

$$s = (1 - p)\, q \qquad (5.7)$$

The maximum *cwnd* ($cwnd_{\max}$) that SCTP will use in slow-start mode is dependent on the value of the slow start threshold parameter (*ssthresh*) that is itself dependent on implementation and configuration details. The SCTP specification [SX00] recommends that *ssthresh* be initialised to the value of the advertised receiver window for the association. We make the reasonable assumption here that it takes a value that is both a multiple of the MSS for the route and a power of two. A typical value would be 64kB for TCP but much larger value is possible for SCTP due to the doubling of the field used to carry the *a_rwnd* value.

In general:

$$w(2^j) = \frac{s^{2^j - 1}}{\sum_{i=0}^{j\max - 1} s^{2^i - 1} + \frac{s^{2^{j\max} - 1}}{1 - s^{2^{j\max}}}} \qquad (5.8)$$

Where $w(2^j)$ is the probability that that $cwnd = 2^j \quad j \in \{0,1,..., j\max\}$ and $j\max = \mathrm{Log}_2\, cwnd_{\max}$

Now the average window size can be calculated from the expectation value of this system:

$$\bar{w} = \sum_{j=0}^{j\max} 2^j\, w(2^j) \qquad (5.9)$$

Thus we can model a single SCTP source operating in slow-start mode across a single non-congested route. This allows us to calculate the average window size (and hence throughput and latency from (5.3) and (5.4)) on that route given the packet loss probability on that route and the average data size to be sent on the route. We must now address how these two models can be combined to give a model of a multi-homed SCTP endpoint.

5.2.3. Modelling a SCTP Multi-Homed Endpoint

For multi-homed endpoints, SCTP establishes an association between two endpoints with multiple network addresses. One address is always considered the "primary" and user data transferred in an

association is normally sent to it. In the case of encountering errors (lost packets) SCTP will use another "secondary" address as the destination for the data. Depending on the precise IP routing dynamics configured this can result in independent paths or links between the endpoints. Hence when congestion (packet drops) are encountered on the primary path, the SCTP specification recommends that SCTP uses the secondary path for retransmissions.

Thus a first estimate of the throughput of the primary SCTP source can be calculated from (5.3), (5.3a) or (5.3b). However in every case this throughput value includes any data retransmitted due to packet losses. Thus it is known as a throughput rather than "goodput" value i.e. it is an estimate of the actual load on the underlying IP network rather than the data rate achieved by applications.

By considering the case where there is just one secondary path, such as in the standard configuration for SS.7 signalling networks, we can estimate the average amount of data to be sent on the secondary path, $\bar{d}$, as a function of the loss rate, p_p, on the primary and the estimated steady-state throughput, B', on the primary.

$$\bar{d} = \bar{B}' p_p \qquad (5.10)$$

As the retransmitted traffic is no longer carried on the primary path we must also modify our estimate of traffic on the primary path, $\bar{B}_p$ as follows:

$$\bar{B}_p = \bar{B}' - \bar{d} \qquad (5.11)$$

These estimates are based on normal operating conditions and take no account of SCTP changing primary path due to very heavy congestion on the primary. However the SCTP path changeover mechanisms are very unlikely to be triggered under normal network conditions with the default SCTP parameters. A more complex characterisation of SCTP multi-homed endpoints can be found in [Pad00] however Padhye et al. show that for normal operating conditions ($p_p < 0.05$) the additional complexity adds little in accuracy while increasing computational complexity. If desired the expressions for throughput developed by Padhye et al. could be substituted in the work that follows. In this analysis we continue to use the simpler expressions (5.3, 5.3a) without loss of generality.

The expressions (5.10) and (5.11) provide the basis for coupling the pair of SCTP sources making up a multi-homed SCTP endpoint. We can now calculate overall characteristics of the system. For the primary path we can calculate the average throughput ($\bar{B}_p$) and latency ($\bar{l}_p$) from (5.2), (5.3), (5.4) and (5.11):

$$\bar{B}_p = \frac{\sqrt{\frac{2}{p_p}} m_p}{\bar{t}_p} (1 - p_p) \qquad (5.12)$$

$$\bar{l}_p = \frac{\bar{s}_p}{\bar{B}_p} \qquad (5.13)$$

Where all parameters subscripted with a *p* have their usual meanings but are specific to the primary path.

For the secondary path we can calculate the average throughput ($\bar{B}_s$) and latency ($\bar{l}_s$) from (5.3) and (5.9)

$$\bar{B}_s = \frac{m_s \sum_{j=0}^{j\max} 2^j w(2^j)}{\bar{t}_s} \qquad (5.14)$$

where $w(2^j)$ is given by (5.7) and we can calculate *s* from (5.5), (5.6) and (5.9) as follows:

$$s = (1 - p_s)\left(1 - \frac{\bar{t}_p}{\sqrt{\frac{2}{p_p} m_p p_p}}\right) \qquad (5.15)$$

Where all parameters subscripted with a *s* have their usual meanings but are specific to the secondary path. The latency on the secondary path, from (5.9), is then simply:

$$\bar{l}_s = \frac{\bar{s}_s}{\bar{B}_s} \qquad (5.16)$$

Thus we have developed a model of the behaviour of a multi-homed SCTP endpoint in terms of two related traffic sources, the primary and secondary. Given the route characteristics of the observed packet loss rate for the primary and secondary route we can predict the average throughput (or load on the IP infrastructure) on each route and the average latency experienced by SCTP applications. In § 5.5 we evaluate the accuracy of the predictions of this model against both simulation and lab-based SCTP experiments.

5.3. Event-Based Simulation Model of SCTP

This model is based on an implementation of the SCTP specification within the OpNet [Opnet] event–based simulation-modelling environment. The SCTP protocol machine was implemented as a protocol layer within the existing OpNet IP modelling architecture. The design patterns established by existing OpNet transport protocol layer models, especially the TCP model, have been followed to implement the

SCTP model. Thus the protocol layer model is implemented as two types of process, a layer manager and a session process. The resultant simulation model has interfaces to application and network (IP) layer protocols, and operates on individual SCTP packets exactly in accordance with the SCTP specification. The general structure of the model is illustrated in figure 5.3 below. Due to the relative lack of mature implementations of SCTP the model does not mimic any particular implementation but instead follows the specification and best common practice as described by the IETF [SX00]. However in the course of this work several important sub-sets of SCTP functionality were identified and the model has the ability to be configured in one or more of these modes (see below for details).

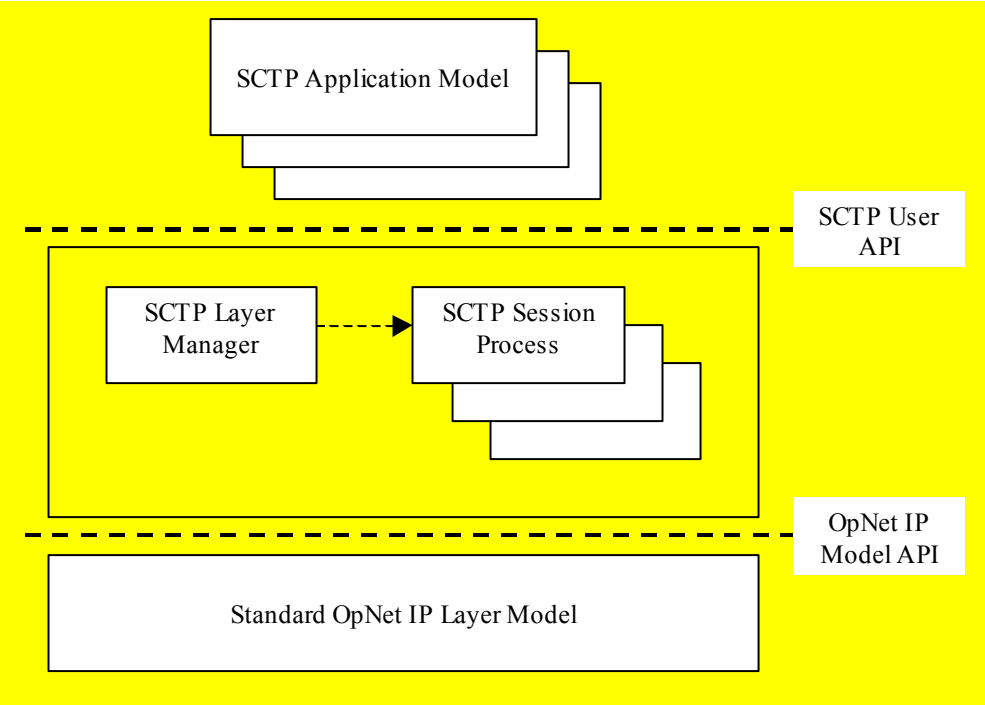


Figure 5.3: OpNet SCTP Model Structure

A general application layer model has also been developed for use with the SCTP transport model, which is configurable as a persistent source or an application making time varying demands on the SCTP transport layer. In combination with standard OpNet models for the IP layer, routers and network links the SCTP model and application models allow a wide range of network analysis functions to be performed.

As illustrated in figure 5.3, the SCTP layer consists of two state machines, a layer manager and a SCTP session process. The layer manager state machine deals with invocations from the application and IP layers and routes them to the correct SCTP session process. It must also handle all incoming SCTP INIT chunks (association creation requests) to implement the three-way handshake for starting a SCTP association. It also maintains a table of "listening" port numbers for local SCTP application processes that have successfully bound to the SCTP layer. The SCTP session process implements the SCTP state

machine as described in the RFC [SX00]. This deals with association initialisation and maintenance, transfer of data and implements SCTP congestion controls. The SCTP application process state machine handles binding to the SCTP layer, variable data send rates, reception of data and logging.

5.3.1. Modelled SCTP Functionality

As is usual with any IETF document a number of features of SCTP have different levels of precedence for conformance to the specification. These are summarized in table 5.1 below along with the features included in our model. Only the most important or relevant mandatory parts of the specification are included in the table below as all mandatory features are implemented in our model.

SCTP Control Feature	RFC	Included In model
Retransmission timer on a per chunk basis	MAY	No[a]
Chunk bundling on transmission	MAY	No[b]
Chunk bundling delay	MAY	No[b]
Delayed SACKs follow RFC2525 section 4.2	SHOULD	Yes
Transmitters are more conservative than congestion control specified in specification	MAY	Yes[c]
Transmitters are more aggressive than congestion control specified in specification	MUST NOT	Yes[c]
SACK reports out of order TSNs in Gap Ack Blocks	SHOULD	Yes[c]
SACK reports duplicate data chunks	SHOULD	Yes
Send a SACK when a packet arrives with both new and duplicate data chunks	MAY	No
Determine if SACK loss is occurring from duplicate TSNs reported in a SACK	MAY	No[d]
Receiver notifies sender in timely manner of changes in ability to send data	SHOULD	Yes
Receiver only increments a_rwnd when data is released from receive buffer	SHOULD	Yes
Receiver puts the current value of a_rwnd in every SACK	SHOULD	Yes
Receiver takes into account that sender will not retransmit data chunks that are acked via cumulative TSN ack	SHOULD	Yes
Receiver does not drop data previously acked in a SACK	SHOULD	Yes
Sender uses rules in specification for calculating rwnd from a_rwnd	SHOULD	Yes
New RTT measurements are made no more than once per round-trip	SHOULD	Yes
Slow-start and congestion avoidance algorithms used	MUST	Yes
Initial value of ssthresh equal to a_rwnd	MAY	Yes
Wait for 3 duplicate SACKs before Fast Retransmit	SHOULD	Yes
Retransmitted data is not used for RTT calculation (Karn)	MUST	Yes

[a] All of the simulations presented here only include one data chunk per packet.
[b] All of the simulations presented here assume that each data chunk represents a maximum size segment so no bundling is possible.
[c] The model has the ability to turn on or off this feature to simulate different variants of SCTP.
[d] No procedures to achieve this are included in the current specification, it is marked "for further study".

Table 5.1: RFC Options Implemented in SCTP Simulation Model

5.3.2. SCTP Variants

In order to investigate the effects of different possible interpretations of the SCTP specification a number of configurations for the SCTP simulation model were possible. The variants used were as follows:

"Standard" SCTP : This variant follows the letter of the SCTP specification exactly and all "MUST" and "SHOULD" features of the specification are supported.

No Gap Ack Block SCTP (NoGap-SCTP): This variant is the same as Standard SCTP but it does not implement the receiver generation of gap blocks in SACKs (these are optional in the specification).

Early Reference Implementation SCTP (RefImpl SCTP): This variant is the same as standard SCTP except that it interprets the fast retransmit procedure in the same way as the earlier (pre version 4.0) reference implementation of SCTP [CM02]. This means that instead of resetting the counter for each TSN gap immediately it reaches 4, it waits until the TSN has been retransmitted. It is to be expected that many early implementations of SCTP follow this reference implementation in its interpretation of the specification.

Modified SCTP: This variant is the same as standard SCTP except that:
1. Once a packet has been retransmitted by the fast retransmit procedure it is marked as ineligible for retransmission again until a transmission time-out has occurred. This more conservative approach is similar to that adopted for various experimental TCP variants [MM96a].
2. It ignores the value of *cwnd* when performing a retransmission due to the detection of packet loss through gap ack blocks. It is a strictly experimental, more aggressive modification to the SCTP congestion control scheme. It also represents a likely interpretation of the current specification by implementers familiar with TCP as in TCP fast retransmit is not constrained by *cwnd*.

5.4. SCTP Congestion Control Evaluation

The final SCTP specification is rather different to the protocol initially envisaged and is not based on a firm foundation of widely available research. Thus it is necessary to begin our evaluation of the SCTP protocol from first principals with an investigation of basic protocol behaviour under a number of error conditions well known in the TCP literature. The basic tool used for this protocol evaluation is our event-based simulation model.

This section presents the results of running our simulation for three scenarios. The two basic scenarios are "Single Packet Loss" - The effects of a single packet loss early in the slow-start phase of a transfer and "Multiple Packet Loss" - The effects of losing 3 packets early in the slow-start phase of a transfer. For each scenario we gathered results for Reno TCP, Standard SCTP, RefImpl SCTP, NoGap SCTP and Modified SCTP. For comparison between Reno and other TCP variants see [Flo96]. In all cases SCTP transfers all data in the association in a single stream of ordered data. This is not an optimal use of SCTP but it makes comparison with a TCP transfer clearer.

Finally in the section "SCTP Bursts" we examine how the loss of a single packet when the congestion window is nearly fully open can produce a burst of packets at the end of the recovery period for certain

SCTP variants.

Note that all of the plots showing packet arrivals at the gateway have been adjusted to show the first packet arrival at a time of 1 second for ease of comparison.

These simulation studies demonstrate that there are three flaws in the current congestion control algorithms:

- Recovery without a timeout after a packet loss is only possible if SCTP generates the optional Gap Ack Blocks in SACK messages. This is because SCTP is only capable of recovering from packet losses by Gap Ack Block reporting or retransmission timeouts. Unlike TCP, there is *no mechanism* for detecting packet loss through duplicate ACK reception.
- The SCTP fast retransmit procedure is vulnerable to being mistakenly triggered multiple times leading to under-utilization of the network during recovery and duplicate retransmissions of the lost packet.
- The SCTP fast retransmit procedure must wait for half a window of data to be acknowledged before retransmitting a lost packet yielding a slow response to packet loss.

We also provide suggested remedies for these flaws that are incorporated into our "Modified" SCTP variant.

5.4.1. Experimental Setup

Figure 5.4 below illustrates the simulated network. The node G indicates a finite buffer gateway, and the end nodes indicate sending (S) and receiving (R) hosts. The queue management strategy in the gateway is drop-tail. The links are labeled with their bandwidth capacity and delay. Each simulation contains an additional flow of non-adaptive FTP/UDP traffic from the sender to the receiver to generate the desired pattern of packet losses. These flows have limited data to send and are not shown in the figures.

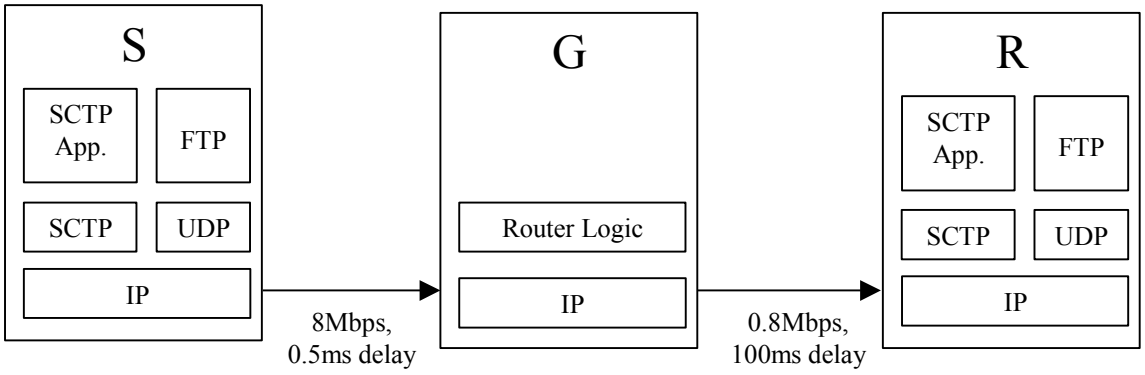


Figure 5.4: Simulated Network

Following the approach of earlier TCP congestion control studies [Flo95, MM96a], the gateway node in these simulations implements a drop-tail congestion control strategy with small buffers. This is not intended as a realistic configuration, just a simple scenario for illustrating SCTP congestion control behaviour. All connections have clock granularity limited only by the precision of the simulation clock.

The simulations all consist of one-way traffic with the transport protocol normally operating in delayed acknowledgement mode i.e. one SACK or ACK for every second data packet received. When packet loss is detected by the receiver it changes to sending one SACK or ACK for every data packet received. Although they are queued at the gateway, none of the acknowledgements are signifigantly delayed or lost. All user data is generated in packets of 1000 bytes. These are transferred in 1000 byte data segments (excluding headers) for TCP and 1000 byte data chunks (excluding headers) for SCTP. For all simulations the initial advertised receiver window was 20,000 bytes (i.e. 20 packets).

The simulation results are shown as a graph plotted at the gateway in a style similar to that employed in [Flo96]. The *x*-axis shows the packet arrival, departure or discard time in seconds. The *y*-axis shows the packet number *mod* 60. Thus a packet arriving and departing the gateway with little delay appears as a single point (■), longer queueing delays produce a resolution of this single point into two colinear points, spaced by the queing delay. A discard is shown with an "X" rather than a point. Acknowledgements are shown as a colinear point (.) one round trip time from the packet they acknowledge.

5.4.2. Single Packet Loss

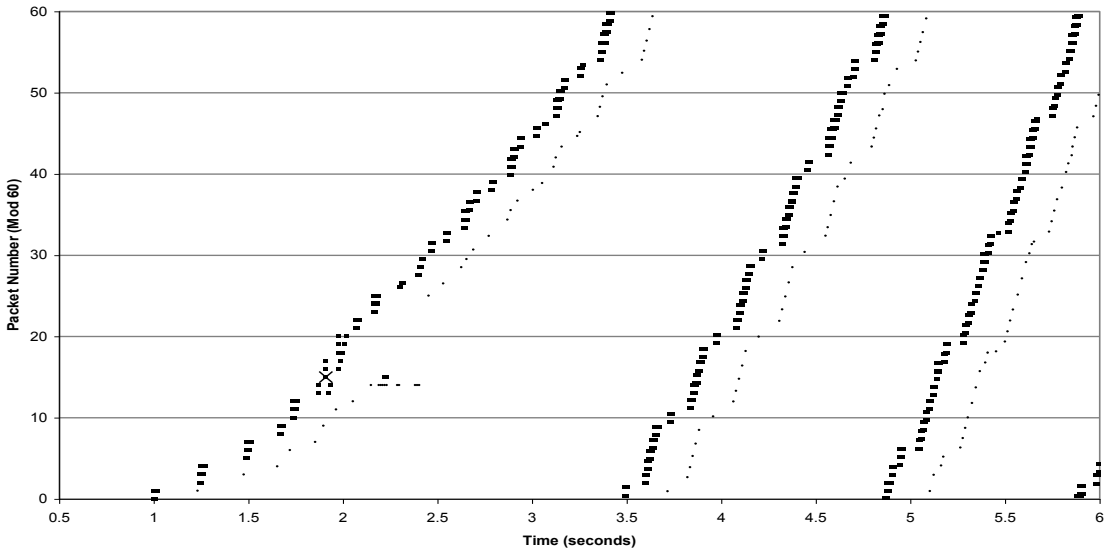


Figure 5.5: Reno TCP with a single packet loss

Figure 5.5 shows Reno TCP's behaviour in this scenario. As shown in earlier studies [Flo96] Reno TCP

handles a single packet loss well and recovers smoothly. Unfortunately the performance of the various SCTP variants is more mixed and is covered in more detail in the subsequent paragraphs.

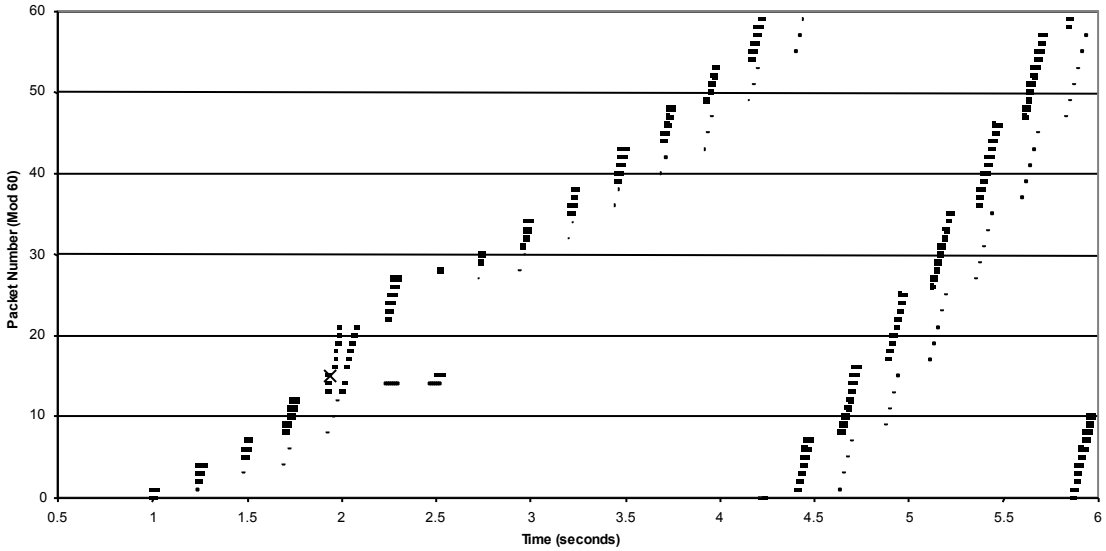


Figure 5.6: Standard SCTP with one packet loss

The behaviour of Standard SCTP (c.f. figure 5.6) after packet 15 is detected as lost by the fast retransmit procedure illustrates three problems in the current specification. These are:

- SCTP must wait for half a window of data to be acknowledged before it can retransmit the packet at approximately 2.5 seconds into the simulation. SCTP reacts approximately 1 RTT later than TCP in this case.
- Even when retransmission is scheduled, the continuous stream of duplicate SACKs arriving at the sender trigger the fast retransmit algorithm multiple times (3 in this case), each time reducing ssthresh and cwnd. The effect of this is noticeable after the SACK acknowledging the retransmitted packet arrives as only small amounts of data are transmitted and increase of the cwnd must be done with the slower congestion avoidance algorithm. This results in a much slower utilization of the available bandwidth than TCP. After the six seconds shown in the plot SCTP has increased cwnd to approximately 14,000 bytes compared with TCP's 15,000.
- Three successive fast retransmissions of packet 15 (at approximately 2.5 seconds) are triggered due to the stream of duplicate SACKs arriving at the sender. These occur as after the packet has been retransmitted once (but not yet acknowledged) the current SCTP fast retransmit algorithm requires that subsequent duplicate SACKs trigger fast retransmit again.

The first two flaws in SCTP congestion control are not inherently dangerous, as they are more conservative than equivalent TCP flows. The third problem is more serious as it requires the sender to inject extra, duplicate data into the network at a time of congestion. Of course all three flaws result in poorer performance than TCP. We propose a simple fix for the multiple triggering of the fast retransmit

algorithm in the current specification that has previously been recommended for SACK TCP implementation [Flo96]. The protocol behavior should be changed so that a packet that has triggered the fast retransmit procedure should no longer be eligible for fast retransmit until a retransmission timeout has occurred. It is also possible to fix the first problem by allowing fast retransmit to take place regardless of the value of cwnd. This second modification is of a more experimental nature and may be too aggressive when competing with TCP flows or in the face of persistent congestion. Nonetheless, the current SCTP reference implementation has adopted both of our recommendations and we provide simulation results for them.

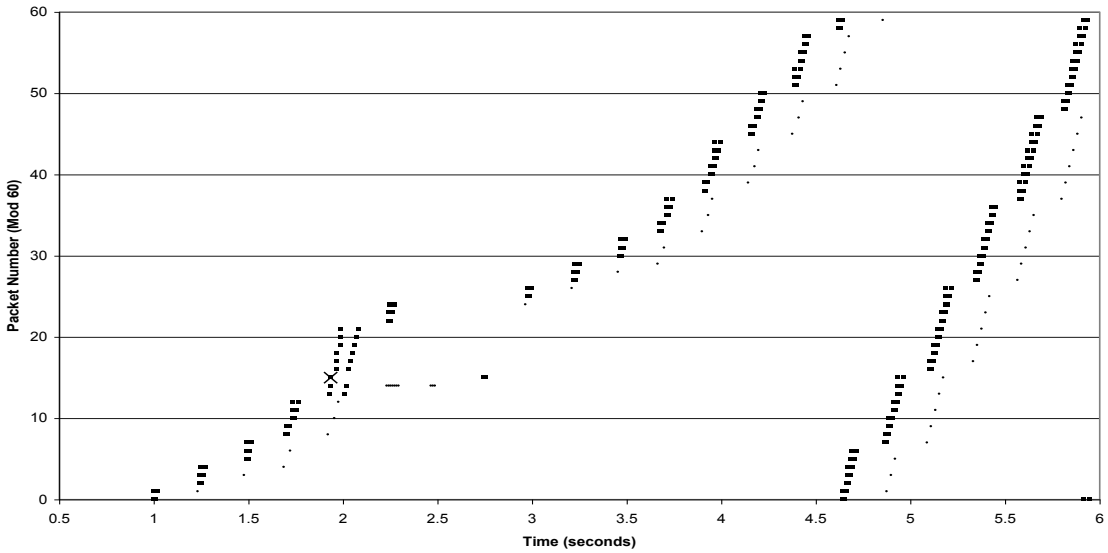


Figure 5.7: NoGap SCTP with one packet loss

Figure 5.7 shows the behaviour of NoGap SCTP. This is even more disappointing and illustrates a further flaw in the current specification. Without gap block generation SCTP cannot trigger the fast retransmit procedure and hence must wait for a retransmission timeout to occur. It then uses slow-start to recover from the timeout. The use of slow-start actually gives almost the same throughput as Standard SCTP due to its use of congestion avoidance from a small value of cwnd. When compared with TCP it achieves a lower throughput but at least it does not mistakenly inject duplicate data into the network. This is obviously a problem with the specification, if a total reliance is to be placed on Gap Blocks then their generation must be mandatory. It would be possible to implement Reno TCP-style controls in SCTP but these have so far proved to be inferior to SACK TCP [Flo96]. This problem is exacerbated by the lack of a gap ack block negotiation mechanism in the SCTP association initialization process. SACK TCP implementations may use the TCP options field to signal their SACK capability at connection start. Another point to bear in mind is that even if a vendor produced a SCTP implementation which did not generate gap ack blocks but was capable of reacting to duplicate SACKs like Reno TCP they could not guarantee that all hosts they communicate with would also have this

capability. In this case the authors recommend that mandatory generation of gap ack blocks for all SCTP implementations is the easiest and most efficient fix.

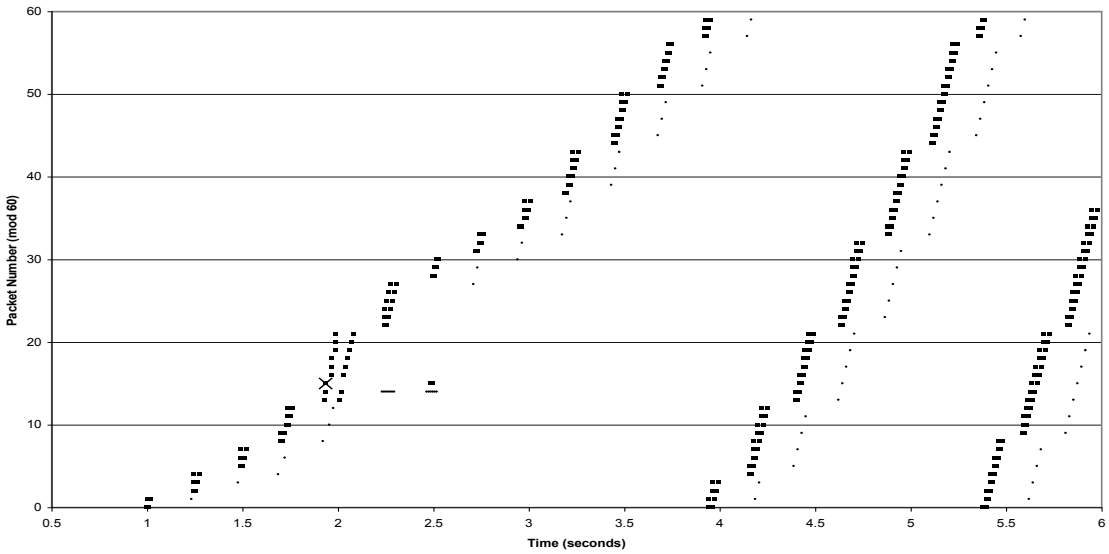


Figure 5.8: RefImpl SCTP with one packet loss

Figure 5.8 shows the behaviour of RefImpl SCTP. This performs significantly better than Standard SCTP but is still inferior to Reno TCP. The reason for the changed behaviour is that when packet loss is detected by the fast retransmit procedure, RefImpl SCTP does not reset the gap counter immediately. Thus subsequent duplicate SACKs arriving increase the gap counter significantly beyond 5. The counter is only reset when the lost packet is actually retransmitted. This protects against multiple triggering of the fast retransmit procedure and hence multiple duplicate packet retransmissions and reductions in ssthresh and cwnd. However as is shown in the next section, "Multiple Packet Loss", it is still possible for this variant to produce duplicate retransmissions. Overall the flow produced by this variant is not dangerous, as it is less aggressive than competing TCP flows. It is however the authors contention that it does not accurately reflect the current wording of the standard and if it is to become the required behaviour then the standard requires some small changes.

Figure 5.9 shows the behaviour of Modified SCTP. This performs better than any other SCTP variant and is very similar to Reno TCP. By allowing immediate retransmission of the lost packet upon detection (fast retransmit ignores the value of cwnd) the recovery process starts as fast as Reno TCP. The introduction of an additional constraint on fast retransmission that packets triggering it can not trigger it again until after a timeout has removed the duplicate retransmission events seen for Standard SCTP. Some slight differences in the figure from Reno TCP can be explained by the TCP model starting to perform some fragmentation of the data segments after cwnd is reduced. Our SCTP model does not currently support fragmentation of user data and hence waits for a full datagram to be allowed

before transmitting. Additionally our SCTP model uses the more aggressive interpretation of the use of cwnd mentioned earlier i.e. if there is even one byte left in cwnd it is permitted to send a datagram (up to the size of the path MTU). In general this variant has been shown to perform well in this case and present no danger to competing Reno TCP flows.

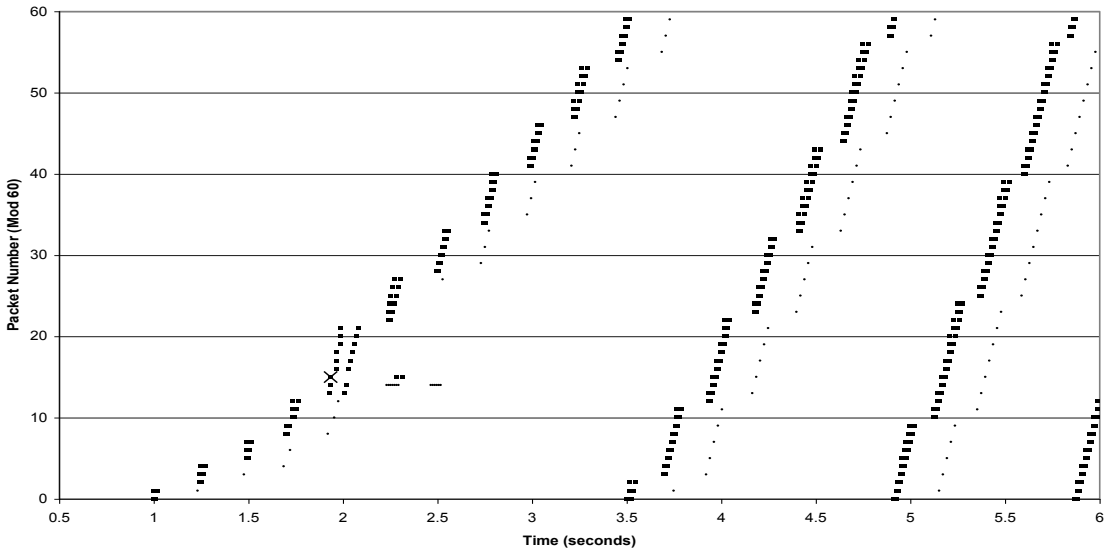


Figure 5.9: Modified SCTP with one packet loss

Figure 5.10 shows the behaviour of Reno TCP when 3 packets are dropped within one window of data. As expected from earlier studies [Flo96] Reno TCP reacts very badly to this situation. This is because it has to wait for the retransmission timer to recover from the dropped packets. This is largely due to Reno's fast recovery behaviour and the lack of an acknowledgement mechanism for out of order data received. Both New Reno and SACK TCP implementations recover more gracefully from this situation.

5.4.3. Multiple Packet Loss

Figure 5.11 shows the behaviour of Standard SCTP for this scenario. It recovers much more successfully than Reno TCP as would be expected by comparison with SACK TCP in earlier studies [Flo96]. This is due to the presence of gap ack blocks in the SACK block which report the arrival of out of order data at the receiver. However the recovery is not optimal as the figure shows both multiple retransmissions of data (packet 18 is retransmitted 3 times, packet 20 is retransmitted twice, packet 21 is retransmitted once) and the consequent multiple halving of cwnd and ssthresh (hence entering a congestion avoidance phase). This produces slower recovery than a SACK TCP implementation and is due to the flaws in the SCTP fast retransmit procedure that were pointed out in the last section. As this recovery is slower than SACK TCP it is not very dangerous but it does include the unnecessary duplicate retransmission of data at a time of congestion. It also under utilizes the link during recovery.

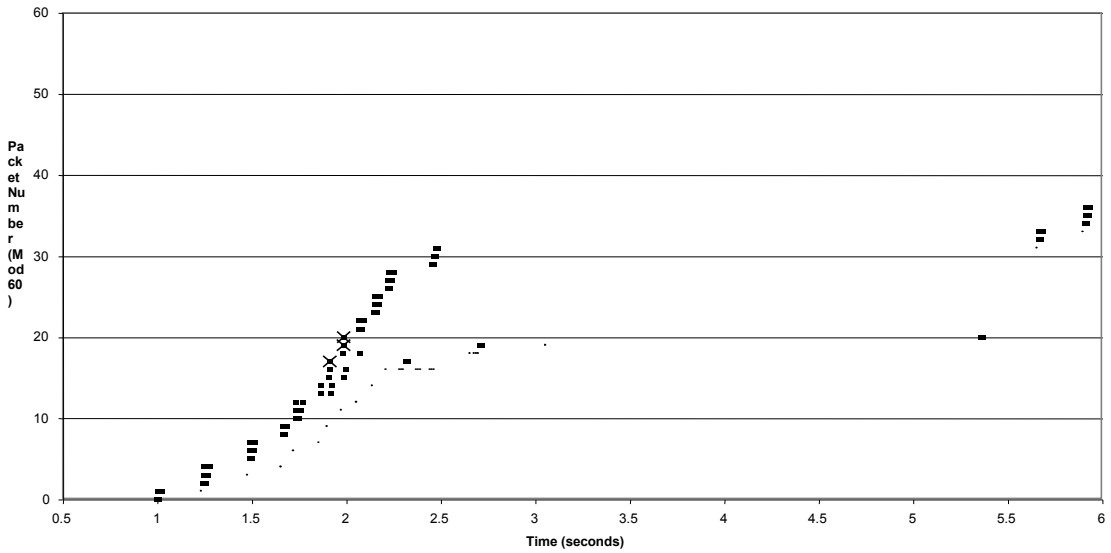


Figure 5.10: Reno TCP with 3 packet drops

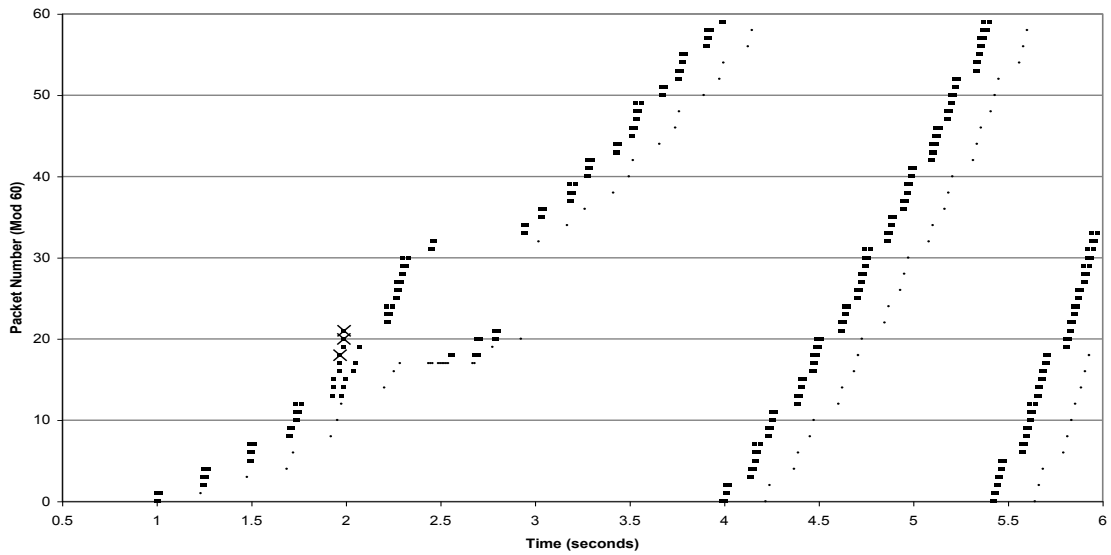


Figure 5.11: Standard SCTP with 3 packet drops

No figure is shown for NoGap SCTP for this scenario as it recovers from any packet loss through a retransmission time out and consequent slow-start of the association (see Figure 5.7).

Figure 5.12 shows the behaviour of RefImpl SCTP for this scenario. As with the results in the last section it performs better than Standard SCTP due to the damping of SCTP's tendency to trigger the fast retransmit procedure multiple times. In this case one additional triggering causes ssthresh and cwnd to be halved unnecessarily. This causes the recovery to be sub-optimal and throughput is reduced. No duplicate retransmission is demonstrated here due to the restriction of cwnd on actual retransmission of data (the acknowledgement for the retransmission arrives before the flightsize reduces sufficiently to

allow the packet to be retransmitted a second time). However it is still possible to get duplicate retransmission with this variant as is shown in the next section.

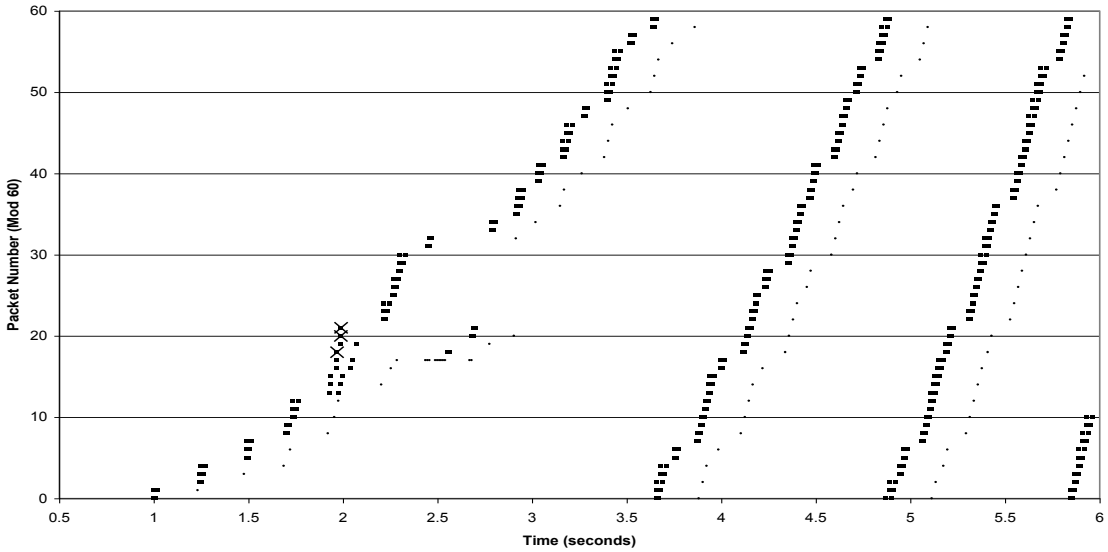


Figure 5.12: RefImpl SCTP with 3 packet drops

Figure 5.13 shows the behaviour of Modified SCTP for this scenario. This variant recovers much more successfully than any other SCTP variant or Reno TCP. This is due to the faster retransmission of data (and consequent reduction in the number of duplicate SACKs received) and this variant's restriction on multiple invocation of the fast retransmit procedure. No unnecessary reduction in ssthresh and cwnd takes place and hence a greater throughput is maintained than RefImpl SCTP.

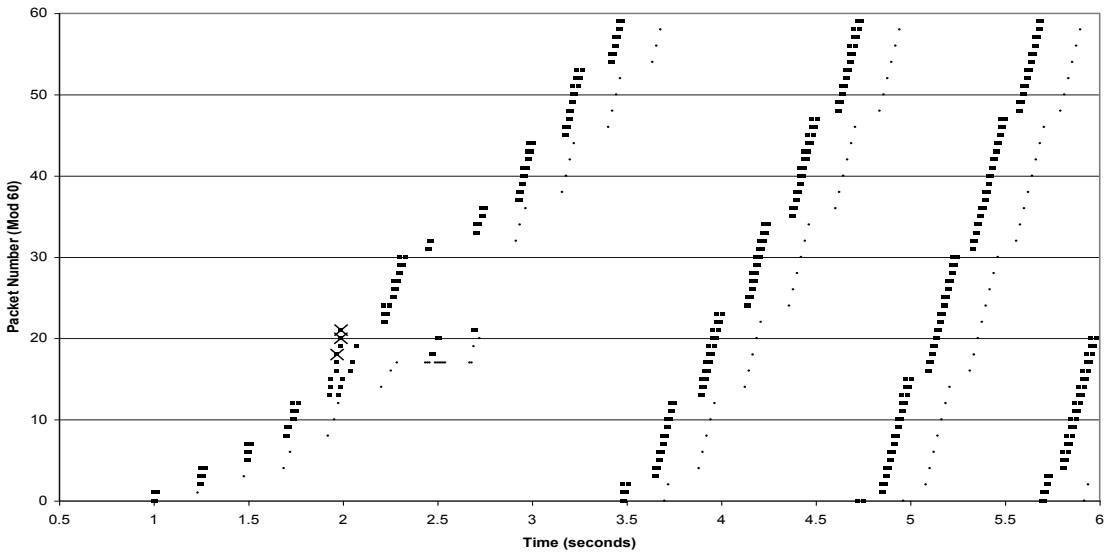


Figure 5.13: Modified SCTP with 3 packet drops

5.4.4. SCTP Bursts

The earlier scenarios demonstrated Reno TCP and SCTP variant recovery from packet loss early in the slow-start period. The window had only opened to approximately 10,000 bytes before packet loss occurred. In this section we examine the behaviour of various SCTP variants when the window is allowed to open almost to its maximum of 20,000 bytes and packet loss occurs. Without packet loss the gateway buffers are sufficiently large to barely sustain the whole transfer of data (based upon the delay-bandwidth product rule). However the behaviour of some SCTP variants when exposed to packet loss with a larger window is to produce a harmful burst of data which itself can induce one or more packet loss events. In all cases packet number 33 is dropped by the gateway due to the background flow not illustrated in the figures. Neither Reno TCP or NoGap SCTP produce any harmful behavior in this case and hence plots for their behaviour in this scenario are not shown.

Figure 5.14 shows the behaviour of Standard SCTP in this scenario. No burst is generated by Standard SCTP, but its performance is very poor when compared with its recovery from a single packet loss earlier in the slow-start phase (Figure 5.6). This is due to the additional numbers of duplicate SACKs present in the network when the loss occurs. Packet 33 is retransmitted 4 times with consequent effects on ssthresh and cwnd. The effect of this single loss is a collapse of cwnd and ssthresh down to their minimum values of 2*pathMTU. This behaviour demonstrates the increasing impact on performance of packet losses with larger window sizes for Standard SCTP (especially in the case of high bandwidth, long delay links, which are often experienced [All99] on the Internet).

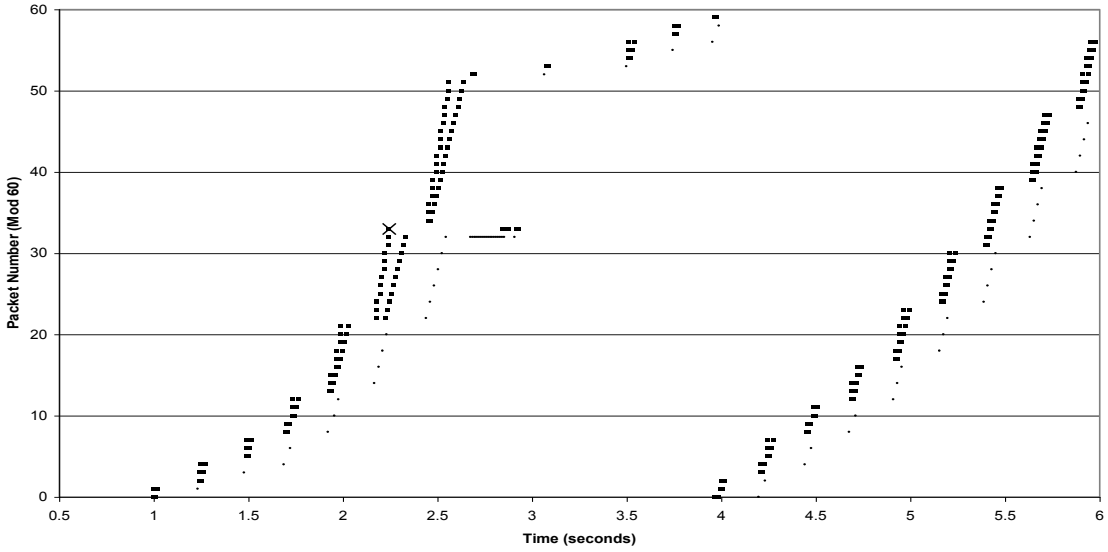


Figure 5.14: Standard SCTP with 1 drop for a large window

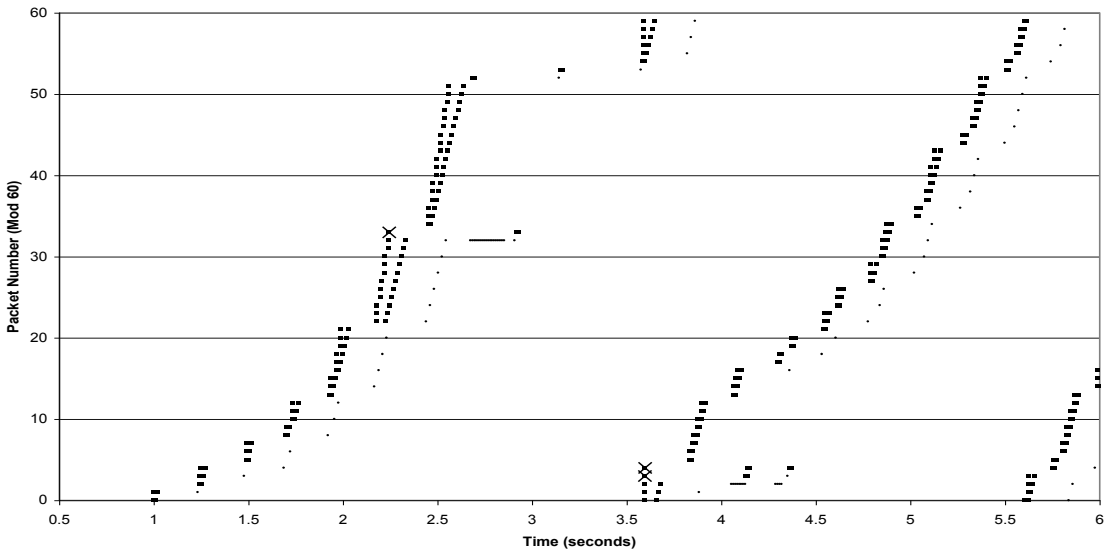


Figure 5.15: RefImpl SCTP with 1 packet drop and a large window

Figure 5.15 shows the behaviour of RefImpl SCTP in this scenario. Here we see the generation of a burst of traffic after recovery from the initial packet loss. This is due to the reduced occurrence of fast retransmit with this variant. At the time that packet 33 is lost cwnd is approximately 19,000 bytes. This value is halved upon packet loss detection. By the time detection has occurred almost a full window (20,000 bytes) of data is in flight. This is all buffered at the receiver as it is out of sequence. This reduces the receiver's a_rwnd to approximately 2,000 bytes. Thus when the SACK acknowledging the retransmission is received by the sender it has a very low a_rwnd and this rather than cwnd constrains transmission of new data. As can be seen in the figure only one new packet is sent in response to this SACK. However the SACK also fully acknowledged (moved the cumulative TSN ack point above) all of the out of order data previously received. This reduces the sender's view of the amount of data in the network to approximately 1000 bytes. When a subsequent SACK arrives (the receiver sends one when it clears its inbound buffers by sending all the previously queued data to the application layer) it will have a full a_rwnd (20,000 bytes) available. The sender is now limited by Min(cwnd, a_rwnd – flightsize) which is approximately 10,000 bytes, the value of cwnd. This produces a burst of packets from the sender, which causes packets at the gateway to be dropped. One interesting thing to note about the recovery of RefImpl SCTP from these secondary losses is that they induce multiple retransmission of packet number 65. This is because even though this variant reduces the number of times fast retransmit is invoked, it still allows multiple triggering of the procedure for a given packet. This of course also unnecessarily reduces ssthresh and cwnd hampering recover from the secondary losses.

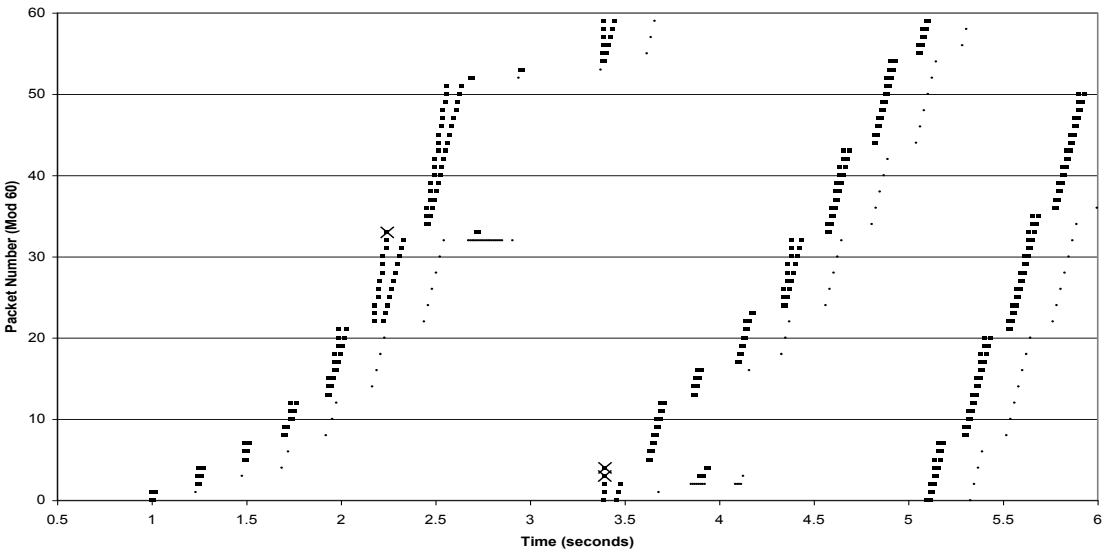


Figure 5.16: Modified SCTP with 1 drop and large window

Figure 5.16 shows the behaviour of Modified SCTP for this scenario. This variant behaves similarly to RefImpl SCTP, producing a burst in response to recovery from the initial packet loss. As with RefImpl SCTP, this occurs on reception of the SACK after the recovery SACK. Due to restrictions on multiple retransmissions no duplicate retransmission takes place and recovery is faster than RefImpl SCTP as cwnd has not been unnecessarily halved. This property of Modified-SCTP makes us hesitate from recommending it to the SCTP standardisation process. However similar problems have been described with SACK TCP [Flo96] and so the use of a *max_burst* control variable (suggested in [Flo96] for SACK TCP) which limited the number of packets SCTP can send in response to a given SACK arrival would be sufficient to control this behaviour.

5.5. SCTP Analytic Model Evaluation

In this section we assess the accuracy and range of applicability of our analytic mode of multi-homed SCTP associations. Comparing both event-based simulation and experimental results with the model's predictions allows us to make this assessment. This section is split into three sub-sections: verification of the congestion avoidance sub-model, verification of the slow-start sub-model and finally verification of the combined model for multi-homed SCTP endpoints.

5.5.1. SCTP in Congestion Avoidance

This is the sub-model used for SCTP primary paths. It has been extensively studied in the literature in the context of TCP connections [MBO99, Pad99, Pad00, MSMO97, Car00]. More recently the current author has demonstrated its utility for SCTP paths in congestion avoidance [BR02]. This is unsurprising given the close relationship of SCTP and TCP congestion controls (c.f. Chapter 2). Given this close

relationship we do not repeat all of these results here. However an example is provided in figure 5.18 which illustrates a comparison of the throughput predicted by the model (5.3b) and simulation results for a variety of SCTP path MTU sizes. Each of the data points on the histogram shows the average relative error for a number of different network route properties for queuing discipline at the router, round trip time and simulation kernel random seed.

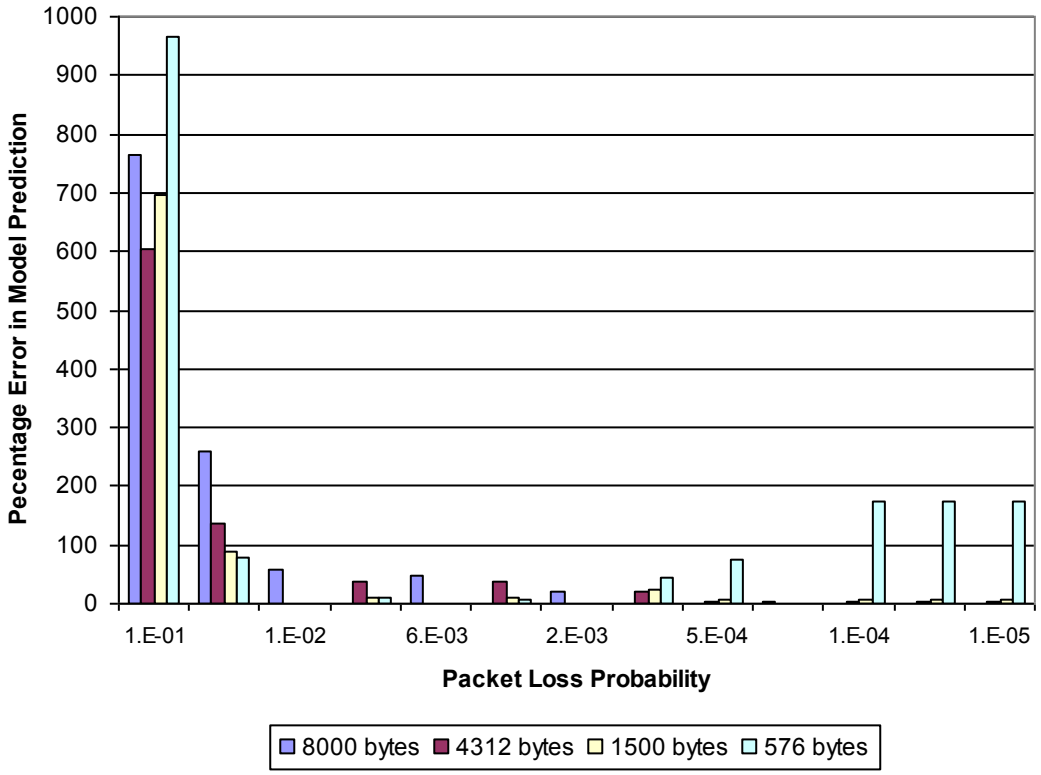


Figure 5.18: Accuracy of SCTP Congestion Avoidance Model Throughput Prediction

The first common characteristic of all the model's predictions is that they are within an order of magnitude of the simulation results. In particular the model is generally more accurate for lower packet loss probabilities where triple duplicate ACK reception dominates over timeout events. A special case in this regard is the series of values recorded for the smallest packet size simulated (576 bytes). This is because most queuing disciplines emphasise packet numbers rather than total queued data size (due to the relative cost of computing metrics based on packet size) whereas the SCTP/SACK TCP congestion controls operate in terms of the amount of data in flight. Thus for smaller packet sizes queuing effects are more dominant on RTTs and hence estimating the RTT as double the sum of link delays becomes more inaccurate.

The second common characteristic of all of the values is that the model over-estimates the throughput achieved. Although it must be noted that when the network is operating at normally observed loss rates ($p \leq 0.005$) [Pad00] that the level of over-estimation is typically 10% or less. When considering

network dimensioning problems (see Chapter 6) this is both an acceptable level of error and a safe direction of error as it leads to more conservative estimates when comparing competing flows. However we must be more cautious when using these values for estimating the network performance observed by applications such as the DIN. A first order correction is to distinguish between the throughput and the "goodput" of a route, where we disregard all of the packets resent due to loss in the network. This is because whereas these resent packets do add to network load they add nothing to the throughput achieved on the route as observed by the application. These topics are dealt with in more detail in §5.5.3, §5.5.4 and Chapter 6.

5.5.2. SCTP in Slow Start

One of the limitations of the source [Eri00] of the TCP slow-start model that has been used to model a SCTP secondary path is a lack of experimental or simulation data that verifies the slow-start model (5.9). Hence, a number of experiments were run using our simulation model of SCTP to verify that SCTP window size during slow start is accurately characterised by the analytic model.

5.5.2.1. Characteristics of the Analytic Model

A matlab program was written to generate a sample of analytic model window size predictions for a range of route packet loss probabilities and mean amounts of data to be sent during slow-start. It is to be noted that the predictions of the model are independent of RTT. In all cases it was assumed that the maximum value of cwnd during slow-start was 64kB, the default value for SCTP [SX00]. This data was then plotted as shown in figure 5.19. In the figure, the window size is given in terms of MSS for the route for which a value of 1000 bytes was assumed. Each plotted line on the figure represents a doubling of the amount of data to send, starting with 1 kB and culminating with 125 kB.

As would be expected, the more data that there is to be sent, the higher the average window size (except for the 125 kB line at packet loss probabilities below 0.006) and as packet loss probabilities go down, the average window size achieved during slow-start goes up (except for the 125 kB line at packet loss probabilities below 0.006). If further plots are performed for higher values of the amount of data to be sent, then the general trend illustrated for the 125 kB line are reproduced, albeit in a more exaggerated fashion. This can be explained by examining our modelling assumptions; If we have a low packet loss probability, then such a route will quickly attain the maximum window size in slow-start (64 kB) and any packet losses from that time onwards will adversely affect the window size of the flow without any possibility of successful packet transmission raising the cwnd further. Thus more damage is done to the long-term average throughput of such a flow than if there was only a limited amount of data to send and the maximum window size was unlikely to ever be achieved. However this does not invalidate our model because it must be remembered that slow-start operation of congestion control will cease once

the slow-start threshold of a 64kB cwnd is achieved. After that point, SCTP will begin to operate in congestion avoidance mode.

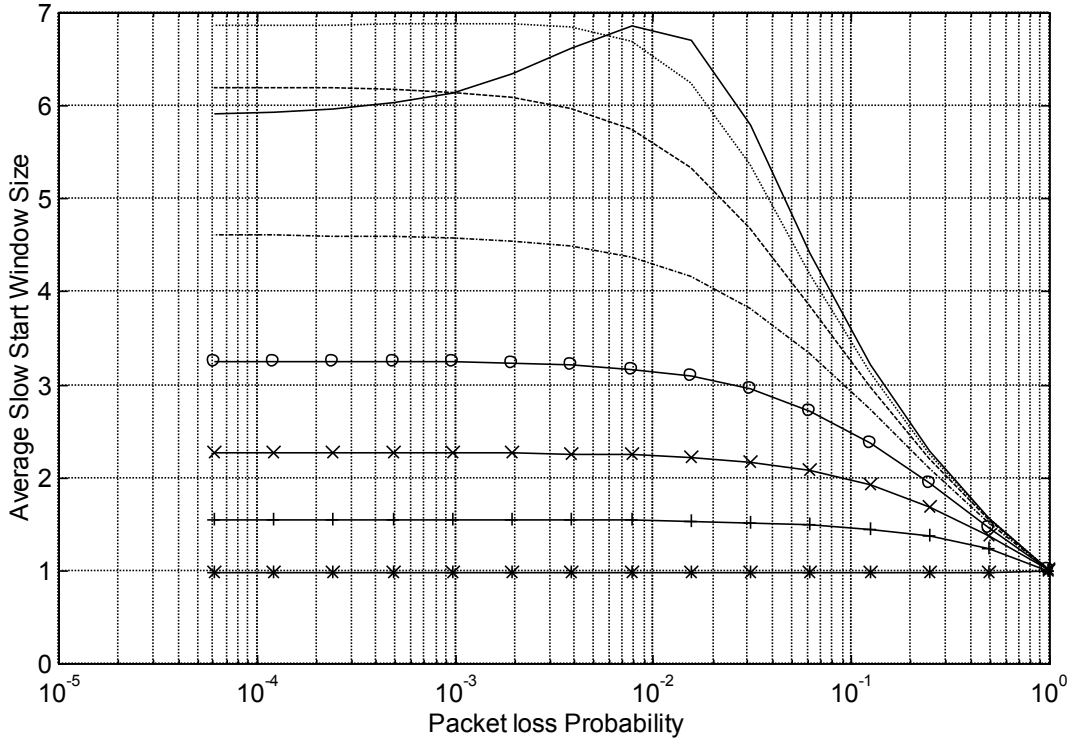


Figure 5.19: SCTP Slow-Start Model Prediction of Window Size

Thus, the observed behaviour can be seen as a bound on the applicability of the model anyway. This fact must be considered when applying our combined model of a multi-homed SCTP endpoint, the modelling assumptions will break down if sufficient data is being sent on our secondary route for it to exit slow-start. This is a function of the loss rate both on the primary and secondary routes.

5.5.2.2. Experimental Verification of Analytic Model for Constant RTT

In the last section some insight was gained into the predicted behaviour of the slow-start analytic model. In this section we evaluate the accuracy of the predicted results by comparison with results generated by the SCTP simulation model. The same experimental configuration as defined in §5.5.1.1 was used with the addition of variable (configurable) packet loss rates at the router and variable transmission delays on the route.

Figure 5.20 illustrates both the analytic model predicted window size (dashed line) and the observed average window size from 1000 runs of the simulation model (solid line). There is a line on the plot for both the analytic prediction and the simulation model sending 4k, 8k, 16k, 32k and 64k of data. The

simulation model plots do not include packet loss probabilities above 0.065. This is due to the difficulty of simulating in this regime as the SCTP protocol will break down due to excessive losses, it is also more than an order of magnitude above the typical loss probabilities observed in real networks [Pad00].

For each run of the simulation model the average window size was calculated by observing the amount of time spent at a given value of cwnd from the start of the transfer to the time at which cwnd returns to the idle value of two maximum segment size PDUs. No plots were included for 1k and 2k transfers as these show very little variation in size from the initial window size and are hence flat horizontal lines. This is because the SCTP slow-start window growth mechanism has no time to be activated when only one or two packets are sent. No plots were included for larger amounts of data to send than 64k. We have already seen in §5.5.2.1 that the analytic model breaks down in that region (and the connection is likely to no longer be in slow-start mode anyway).

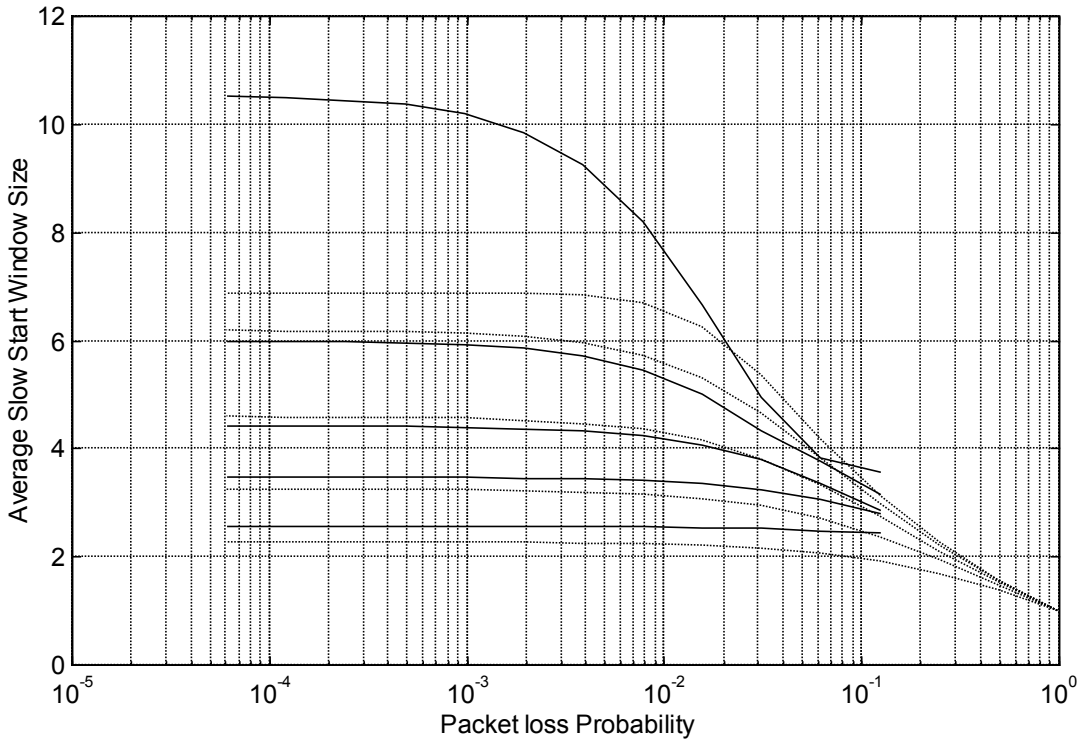


Figure 5.20: SCTP Slow-Start Model Compared with Constant RTT Simulation Results

From the figure it can be seen that the model predictions are very accurate (generally within 10% of the experimental value), especially in the likely operating region of SCTP networks ($p = 5 \times 10^{-3} \pm 4 \times 10^{-3}$) , except for the case where there is 64k of data to send. In this case the analytic model predictions give a value that is between 90% and 60% of the simulation experimental results. This degree of accuracy is within that required for dimensioning procedures but it is unfortunate that the model under-estimates

window size (and hence network load) in this region. Depending upon the application, the analytic model may thus be considered useful for scenarios where more than 32k of data is to be sent, or as in the case of 128k or larger payloads another model may be applied.

A trend evident in the figure is that the analytic model under-predicts slightly the window size for small amounts of data to be sent (4k, 8k). This is probably because the model assumes that the initial window of the connection is 1k (i.e. 1 MSS). It may be thought that a model that more closely resembles the physical situation of starting at 2k (i.e. 2 MSS) would improve the results in this region. The resultant Markov chain with states at cwnd = 2k, 4k, 8k, 16k, 32k and 64k is easily solved in a manner similar to that of §5.2.2 and the resultant estimates of window size are provided below in figure 5.21 for comparison with figure 5.20. Unfortunately, as can be seen from the figure, this solution over-compensates, as it does not take into account the real possibility of the window slipping back to 1k due to multiple losses. In all cases this model would over-estimate the average window size and it is definitely not an improvement over the model presented in §5.2.2 and figure 5.22.

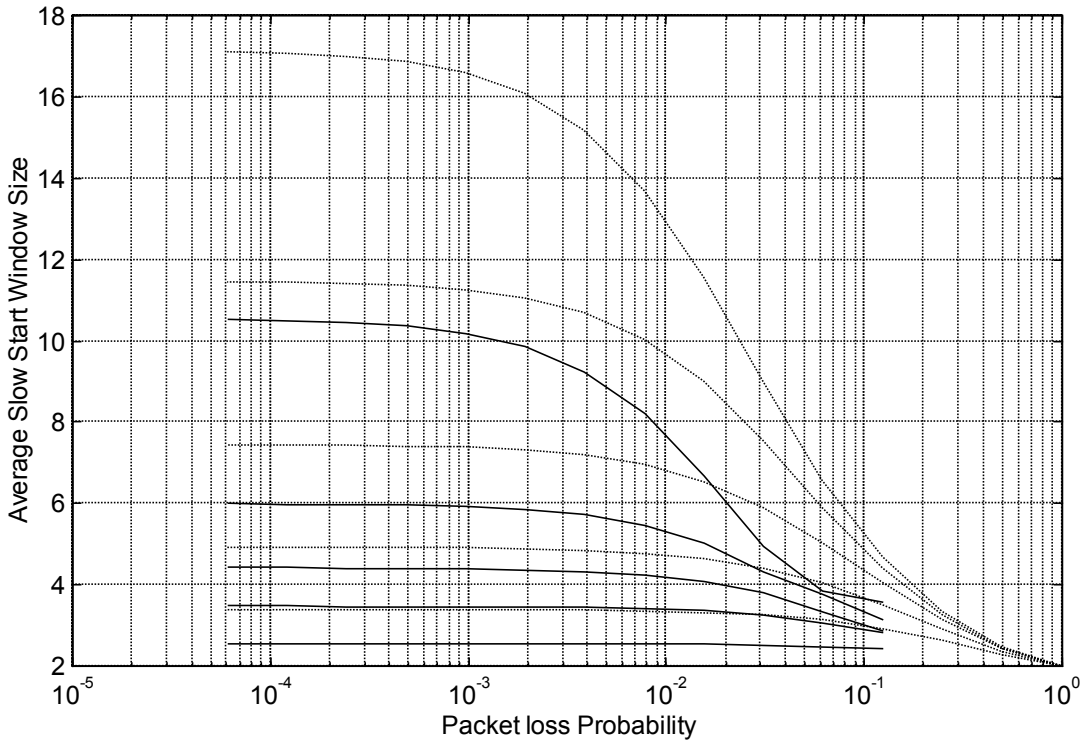


Figure 5.21: Modified SCTP Slow-Start Model Compared with Constant RTT Simulation Results

5.5.2.3. Experimental Verification of Analytic Model for Variable RTT

The analytic model predictions can be seen to be RTT independent, thus a priority issue to investigate is the effect (if any) of varying RTT. The method adopted here is to compare the behaviour of a 32k

transfer (corresponding to one line on figure 5.18) with simulation results for a range of configured round trip times. The predictions for average window size given 32k to transfer were chosen because in §5.5.2.2 we have shown that the model gives good predictions in this region and also because with larger amounts of data to send the stochastic aspects of our simulation model produce more variation in the possible results for a given simulation run. All experimental results are based on the average of 1000 simulation runs.

Figure 5.22 below illustrates the behaviour of this system. The dashed line corresponds to the RTT independent analytic model prediction of average window size and each solid line corresponds to the observed simulation behaviour for a different RTT. The simulated RTTs plotted are 0.01s, 0.02s, 0.04s, 0.08s, 0.16s, 0.32s, 0.64s and 1.28s. These values correspond to the likely operating region for SCTP RTTs for LANs or engineered wide-area networks.

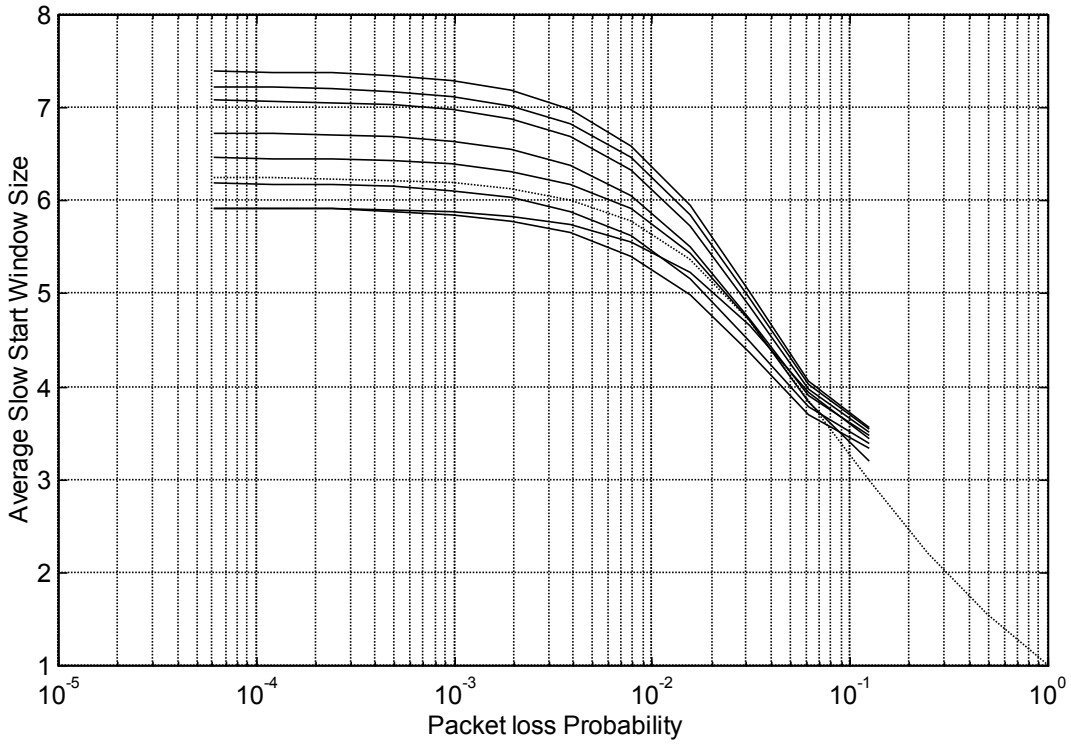


Figure 5.22: SCTP Slow-Start Model Compared with Variable RTT Simulation Results for a 32k Transfer

The highest average value of window size is given by the lowest RTT and from RTTs of 0.01 to 0.32s each doubling of the RTT drives down the maximum average window size by an average of 4.58%. Above a RTT of 0.32s doubling the RTT has less of an effect on the average window size and the values are clustered together. This is likely to be a result of lower RTTs resulting in faster recovery from retransmission timeouts and consequent increases in the average cwnd over the simulation time. In

all cases within the likely operating region of SCTP network losses ($p = 5 \times 10^{-3} \pm 4 \times 10^{-3}$) the simulation results are within 15% of the value predicted by the analytic model. This level of accuracy is sufficient to have confidence in the predictions of the analytic model over varying network round trip times. Plots for transferring other amounts of data show similar results.

5.5.2.4. Experimental Verification of Analytic Model for Limited Bandwidth

Slow-start window size evolution as characterised by the analytic model assumes that there is unlimited bandwidth available to the SCTP source. As sources in slow-start typically only have small amounts of data to send (and do so at a conservative rate) this assumption is generally valid. Here we test the limits of this assumption by plotting both the predicted and simulated average window size for a 32k transfer for varying access link bandwidths with a fixed RTT of 0.02s. As in §5.5.2.3 all experimental results are based on the average of 1000 simulation runs.

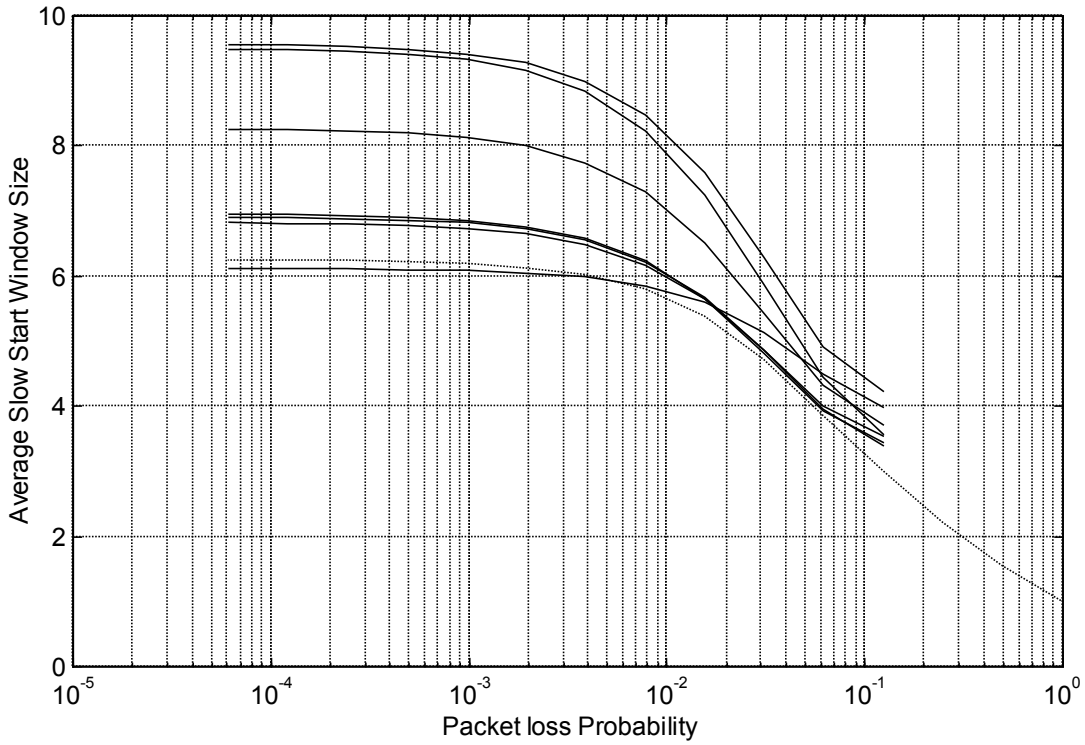


Figure 5.23: SCTP Slow-Start Model Compared with Variable Bandwidth Simulation Results for a 32k Transfer

Figure 5.23 illustrates the results of these experiments. In the figure the dashed line shows the bandwidth independent analytic model prediction, the solid lines show the bandwidth limited simulation results. Lines are plotted for access link bandwidths of 5kbps, 10kbps, 20kbps, 40kbps, 80kbps, 160kbps and 320kbps. The three largest values of average cwnd shown are associated

respectively with the 5, 10 and 20 kbps plots.

5.5.3. Multi-Homed SCTP Experiments

Given our evaluation of the slow-start model (§5.5.2.2) and previously published evaluations of the congestion avoidance model [Pad00, MSMO97], the final stage in our model evaluations must be to estimate the accuracy of our approach modelling for multi-homed SCTP endpoints (§5.2.3).

5.5.3.1. Experimental Set-up:

An experimental network similar to that outlined in [JST00] was established. This network consisted of a Wide Area Network (WAN) emulator ("NIST Net" [NISTNet]) connected to two LANs. Each LAN had as a client one multi-homed SCTP endpoint (a PC running Linux OS, the Reference Implementation of SCTP [CM02] and two network adapters). NIST Net allows the configuration of packet delay distributions, congestion and background loss, bandwidth limitation, and packet reordering / duplication at the router. Each route through the emulated WAN may have different characteristics. The physical network configuration is illustrated in Figure 5.24.

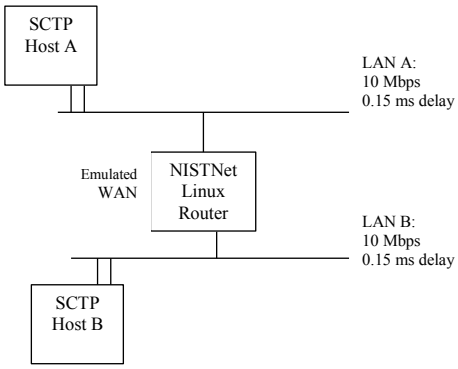


Figure 5.24: Experimental Set-up

In order to enable data collection for the experiments, the "Reference Implementation of SCTP" protocol stack was modified to include some instrumentation functionality. In addition the Ethereal [Par98] network protocol analyser software was installed on the router to collect additional information.

This experimental network enabled the exploration of many different network configuration scenarios through manipulation of NIST Net parameters. Figure 5.25 illustrates the capabilities of the network for one possible WAN scenario. NIST Net has been used in the past to test many applications and protocols and is a well-established tool in the Internet protocol development community [Par98].

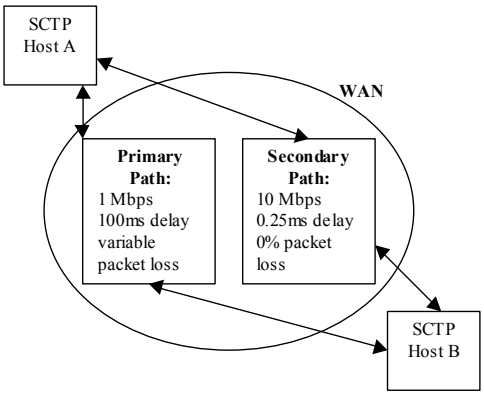


Figure 5.25: Experimental Set-up Capabilities

Establishing a Packet Drop Model

The packet drop scheme used by NIST Net would be critical to the relevance of any results obtained from the experimental set-up. NIST Net provides two types of configurable packet drop models, either a fixed packet drop probability applied to all packets on a given route or the Derivative Random Drop (DRD) packet dropping strategy [Gay96] which is a modification of the RED (Random Early Detection) proactive packet dropping strategy. DRD allows configuration of parameters which induce packet drops based on network load. The NIST Net emulation environment (as opposed to an actual router implementation) allows any undesirable correlation by DRD of packet drops to be eliminated [NISTNet].

Early experiments were carried out with a uniform packet drop probability configured within NIST Net. After this unrealistic configuration was used to verify the basic operation of our experiments, a methodology based on that used in [Pad00, MSMO97] to estimate observed packet loss probabilities in real data was used. NIST Net was configured with DRD in operation to give load-based packet drop patterns similar to those observed in a WAN. In this case, an estimate of the observed packet loss probability can be determined by dividing the total number of loss indications received by the total number of packets sent.

To refine this approach by further decoupling the packet loss events from the transmissions on the primary SCTP path, a background process of ttcp [Par98] generating additional load on the link was added. The same method as above can then be used to calculate the observed packet loss probability.

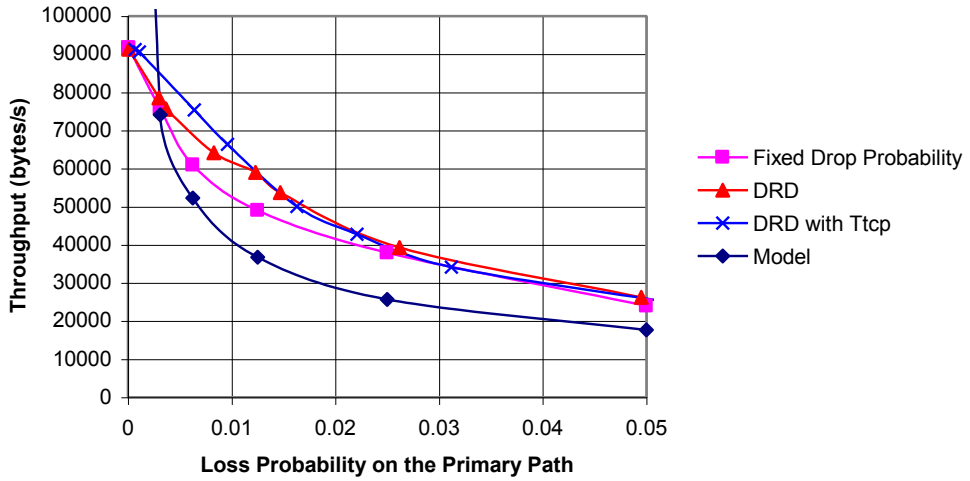


Figure 5.26: Packet Loss Model Effect on Throughput

Figure 5.26 above illustrates the different relationships observed between packet loss probability and throughput for a 2MB transfer on a 150 ms delay primary link for each experimental method and the model predictions. As can be seen from figure 5.25, all of the packet drop generation methods generate fairly similar results, especially as the packet loss probability increases. As the packet drop probability decreases there is less agreement between the observational methods and the fixed probability method. This is due in part to the increased difficulty in accurately estimating the actual packet loss probability from observed packet transmission events for low loss situations. For the purposes of evaluating the approximate solution generated by our analytic model it is significant that all of the experimental methods tend to agree more closely with each other than with the model predictions. The final method (DRD with ttcp) was used for all results presented in the next section.

5.5.3.2. Experimental Results

The throughput model was applied to a range of multi-homed SCTP scenarios, which were recreated in the experimental environment. These scenarios covered primary path loss probabilities from 0.003125 to 0.05 and network delays of between 10ms and 1s. In all cases the primary link was configured with 100kB/s bandwidth and 2MB of data was transferred by SCTP over the association.

An example set of results obtained for a fixed network delay of 150ms and a range of packet loss probabilities is shown in figures 5.27 (primary path) and 5.28 (secondary path) below. In each case the model's primary and secondary path throughput estimates were calculated and plotted against the experimental results.

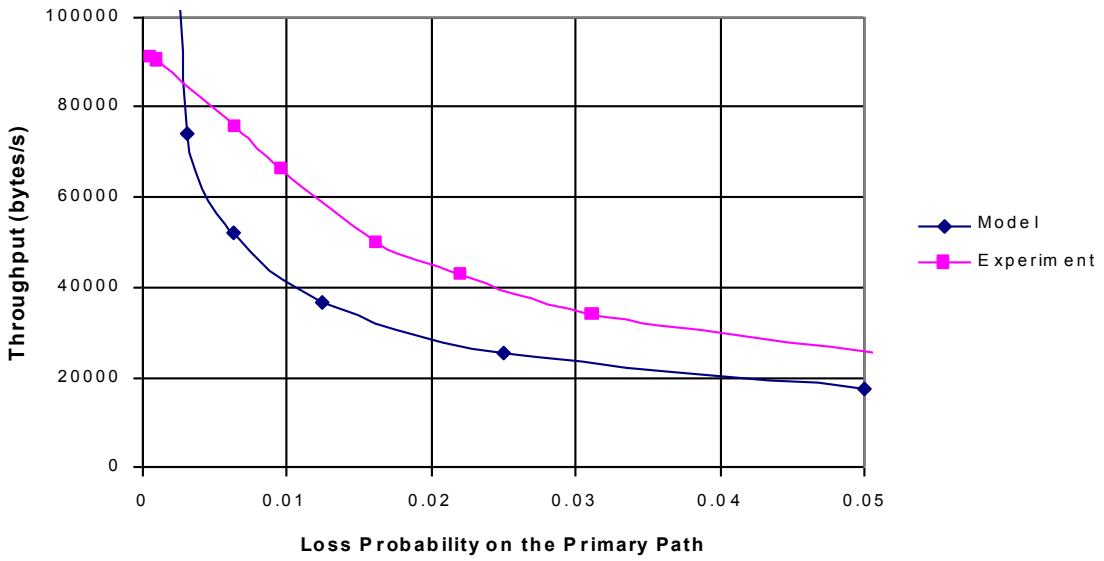


Figure 5.27: Comparison of Model and Experimental Throughput on Primary Path

On both plots we can see that the model's predictions, from (5.12) for Figure 5.27 and from (5.14) for Figure 5.28, give results that quite closely match the trends observed in our experimental results. There is a significant level of error in the model's predictions but given the simplicity of the relations involved this is not surprising. It is our contention that these models are still sufficiently accurate to provide a useful estimating tool. For example, typical margins of error for network planning purposes are within an order of magnitude and our results are well within this range. The plots show that the models generally provide a pessimistic estimate of the throughput exhibited in the experiments. After a limit in the region of p = 0.003, the primary path predictions diverge from our experimental results. In this region it is necessary to use the expression (5.3c) to account for the maximum bandwidth available in congestion avoidance otherwise the theoretical throughput will tend to infinity as packet loss probability tends to zero.

In the region where the estimate of throughput can be considered valid (p > 0.003), the model consistently under-estimates the loss rate on the primary path (and hence the throughput on the secondary path). This is because in the TCP analytic model p is not in fact the loss rate but the probability that a packet will be lost within a TCP "round" given that there has been no losses so far in the round. Once such a loss occurs, the remainder of packets sent in the current round are also considered to be lost. Thus in the TCP analytic model packet losses occur in groups, resulting in an effectively higher per-packet loss rate.

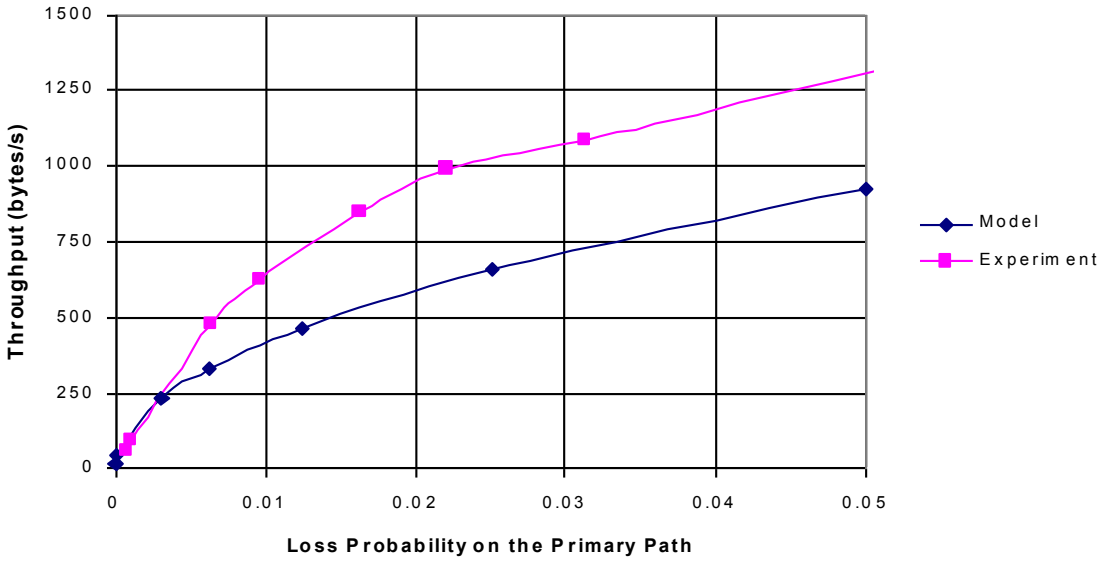


Figure 5.28: Comparison of Model and Experimental Throughput on Secondary Path

Figures 5.29 and 5.30 illustrate a particular network scenario, in fact a large amount of experimental data has been gathered to verify the validity of the model in under different network conditions. Figures 5.29 (primary link model predictions) and 5.30 (primary link experimental results) illustrate the results of this work, summarised as a plot of a three-dimensional surface indicating the relationships between loss, delay and throughput on the primary link. An estimate of the accuracy of the model predictions is presented in figure 5.31 which shows the percentage difference between the predicted throughput values and those measured by experiment.

As can be seen from the figures, there is a good general agreement between the model and experiments over a wide range of values. In general the model predictions are more accurate as loss rates and network delays increase. In terms of model prediction quality there is a problematic region in figure 5.27 where low loss probabilities (< 0.006) are combined with low network delays (< 100ms). However a simple solution to the bad behaviour of our function in this region is to bound its use by the maximum bandwidth available. Thus all solutions in this region are realistically limited. In reality, the loss rate and round trip time would increase dramatically as the throughput approached the capacity of the link, thus our assumptions of fixed delays and loss rates limit the model's accuracy in this region. However in Chapter 6 we see how this approach can be generalised to networks of flows by relating the observed packet loss probability to the offered traffic in the network, thus removing one of the parameters of our model as applied here. Bounding the model has removed a spike in this region in figure 5.27 and replaced it with a flat cut-off at 100 kb/s for the network scenarios examined. This would over-estimate the throughput in this region slightly (~ 3%), but this level of error is consistent with the rest of the

model's predictions. The observed primary rate throughput is on average within ~19% of the predicted value. For the vast majority of observed cases the model under-estimates the traffic by this amount. Considering the wide range of network conditions considered this level of error is acceptable for most applications.

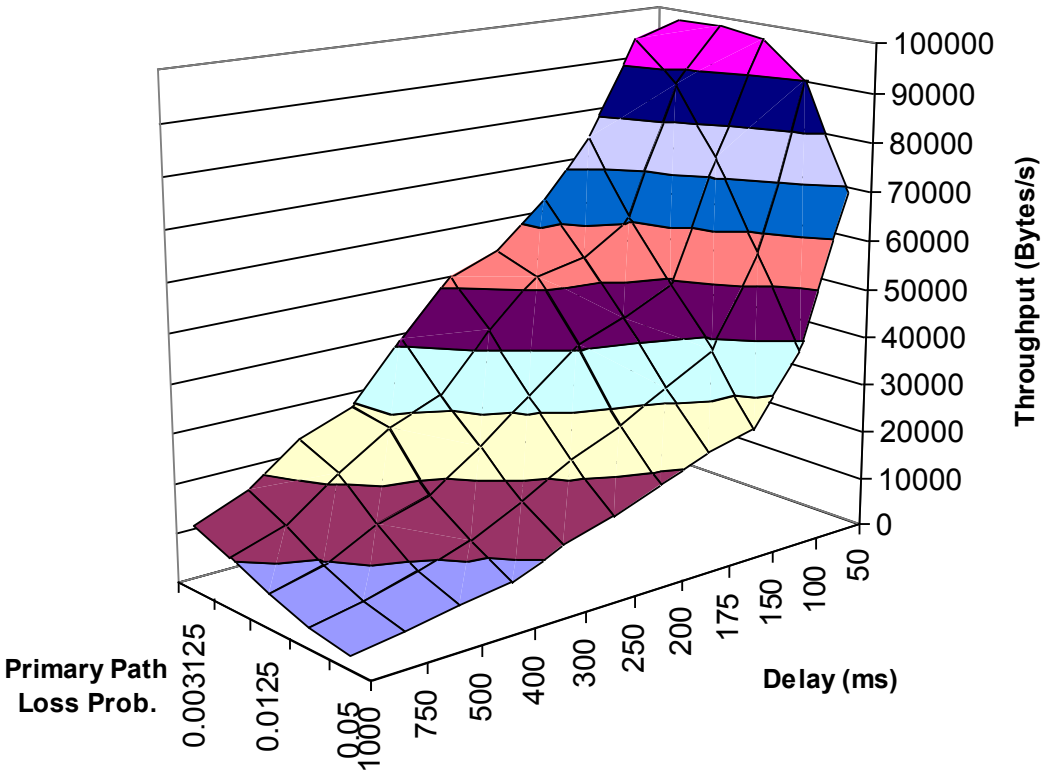


Figure 5.29: Experimentally Observed Primary Path Throughput

In general there is less error observed for networks with higher delays. There also seems to be an area in the plot with minimal errors for packet loss probabilities of 0.025. However most observed percentage error values are very closely distributed.

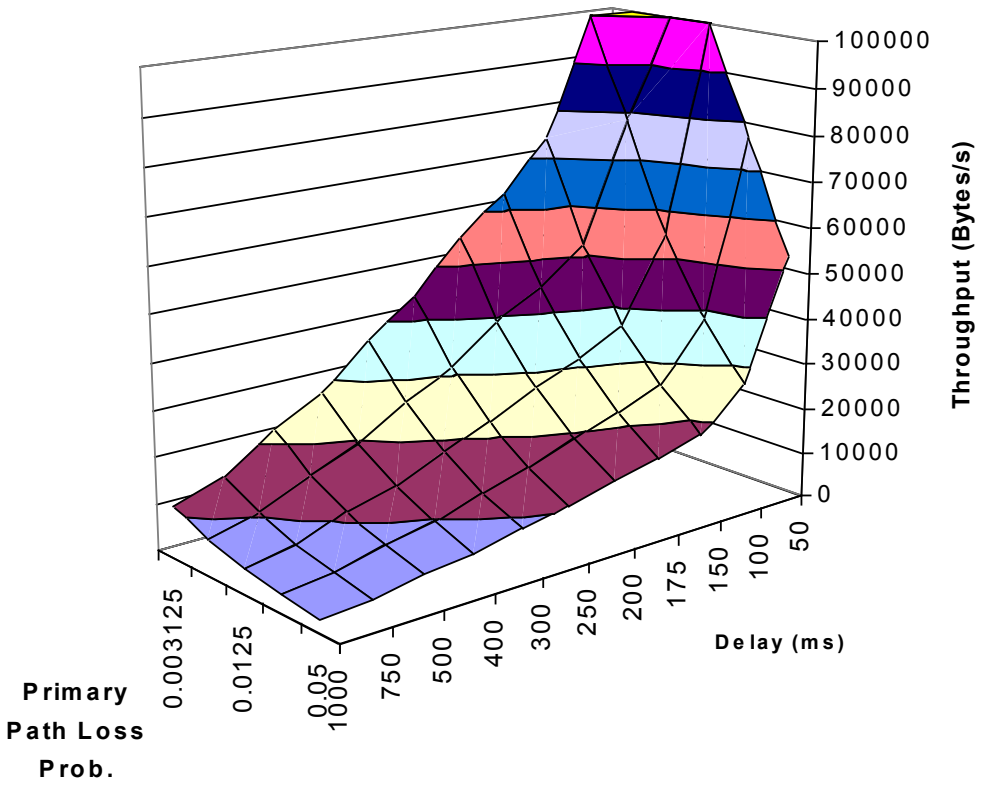


Figure 5.30: Model Predictions of Primary Path Throughput

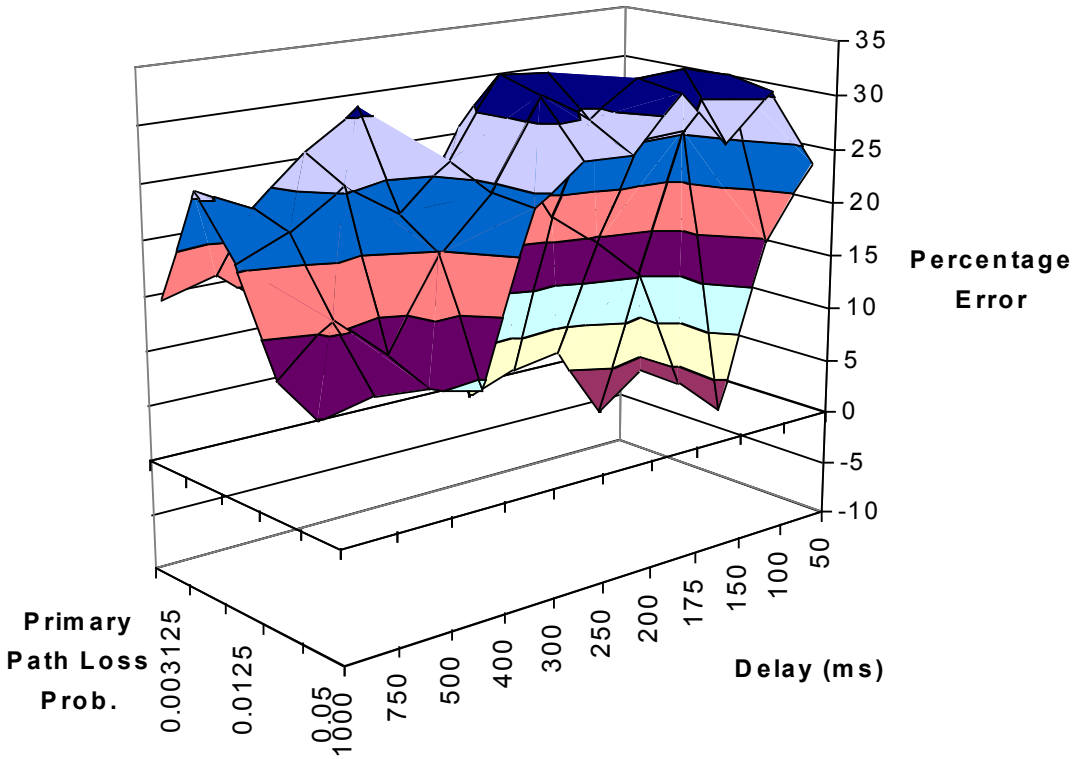


Figure 5.31: Percentage Error in Predictions of Primary Path Throughput

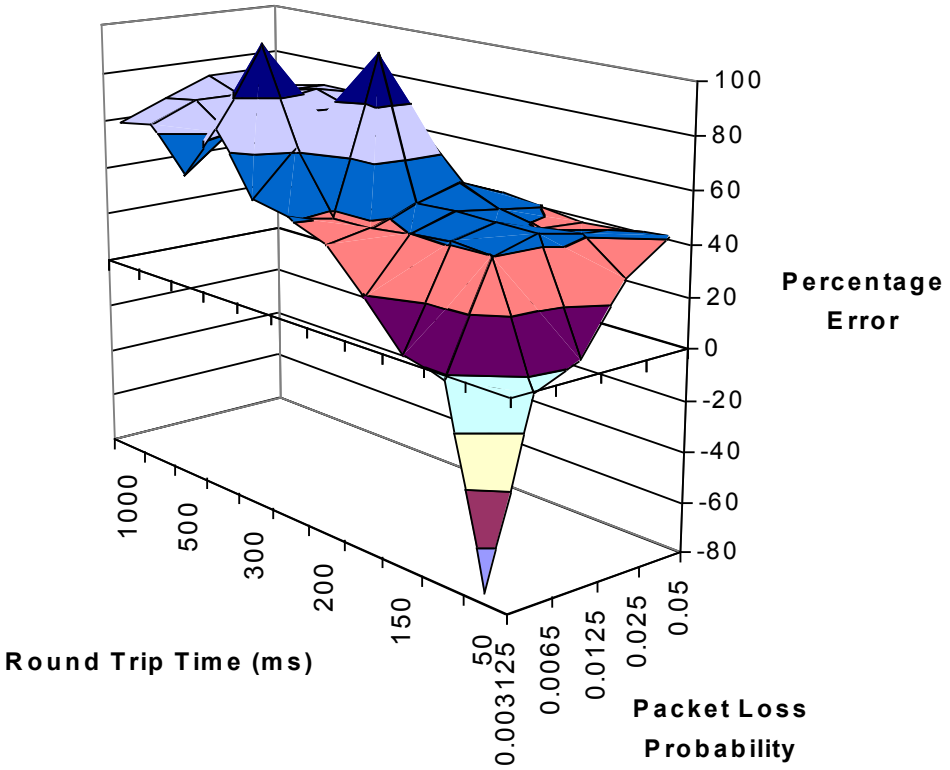


Figure 5.32: Percentage Error in Predictions of Secondary Path Throughput

Figure 5.32 illustrates the percentage error results for the secondary path. It exhibits a much higher overall level of error but the majority of observations are within ±50% of the predicted value (with an average percentage error of 51%). It should be noted that in these experiments the secondary path has a zero packet loss probability. By observing our results from §5.5.2 it can be seen that the secondary path predictions would be even more accurate if there was loss on the secondary path.

The results illustrated here are a reflection of our wider findings i.e. that (5.12) and (5.14) provide a useful 1st-order model of multi-homed SCTP paths. It is important to note that these experimental results represent an independent verification of our model and simulation results as a third party SCTP implementation was used to generate them.

5.5.4. Multi-Homed Model Applicability for Quasi-Persistent Sources

In the previous sub-sections we have primarily dealt with persistent SCTP sources i.e. ones that always have data available to sent to the network. Of course our model makes use of both persistent and non-persistent components to handle the transmission characteristics on the primary and secondary routes respectively. However the application using SCTP has always been assumed to have more data to send. If we take the case of signalling nodes in a network, such as DIN application entities, then it is likely that they will have long-lived SCTP associations between them using the multi-stream abilities of

SCTP to differentiate between individual service sessions. In this scenario it is probable that there will be a relatively large amount of data traffic between the nodes on this single association. However, unless the network is under-dimensioned (and hence over-loaded) it is unlikely that this will correspond to a truly persistent source. In this section we examine how far from our persistence conditions that we can say that our model remains a useful approximation. This will be an important constraint in Chapter 6 when we use our extended model for networks of SCTP flows to estimate the performance of DIN-based SCTP applications.

5.5.4.1. Results

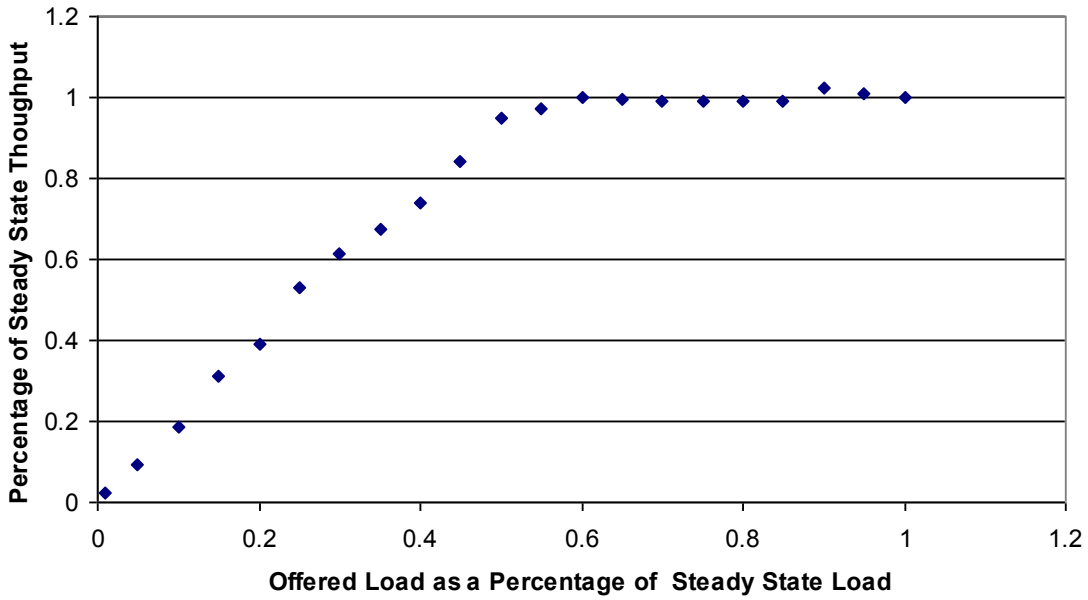


Figure 5.33: Throughput Predictions for Quasi-Persistent Sources

Given a single multi-homed SCTP association we wish to establish the range of application offered loads for which the persistent source assumptions of our model still produce reasonable results. In this analysis we neglect the effects of queuing at the SCTP layer (see Chapter 6 for a treatment of this). Thus we can assume that for offered loads greater than the simulated steady state throughput of the route our SCTP persistent source conditions are met. Hence the region of interest is for offered loads of less than the simulated steady state throughput. Therefore we modify our SCTP application model to implement a Poisson arrival process of SCTP data transfer requests. We can then simulate the effects of varying the mean offered load from the steady state load for a range of network conditions. The results are then reported in terms of the relative error between these simulations and the model's predictions. In all cases competing adaptive network traffic generates all packet losses. This is a worst case scenario from the point of view of maintaining our route's share of network resources and hence for the accuracy

of our model's predictions. We term these applications offering less than the steady state load as quasi-persistent sources.

From figure 5.33 we can see that the steady state model has a wide range of applicability to problems that do not feature truly steady state sources. In fact due to the self-balancing dynamics of competing adaptive sources we see that send rates (offered loads) of down to half the original send rate will achieve very similar throughputs. Once the offered load reduces below this then the throughput reduces proportionately to the reduction in send rate. The complexity of the network and number of competing sources will of course change this dynamic. However the important point to realise is that if our model's approximations are only accurate to within an order of magnitude over some parts of the operating range then send rates up to 90% below the steady state send rate are also appropriate applications of the model. Send rates closer to the steady state rate must also take into account queuing effects at the SCTP layer. This is treated in more detail in chapter 6.

5.6. Summary and Conclusions

In this chapter we have developed and presented a multi-homed SCTP network transport model based on previous work on single-homed TCP transport models and a DIN transport model. In addition to these analytic models, suitable event-based simulation models and experimental configurations to evaluate them have been described. A large number of experiments have been performed to evaluate both the SCTP specification and the accuracy of our analytic models. The main conclusions of this work are as follows:

- **There are flaws in the current SCTP congestion control mechanisms.**
  The current specification of the fast retransmit procedure and the rules for generation of gap ack blocks is weak. It is our recommendation that the generation of gap ack blocks become mandatory in the specification. We have proposed that the fast retransmit algorithm be modified so that a packet that triggers fast retransmit cannot trigger the fast retransmit procedure again until after a transmission timeout event occurs. We have also demonstrated the superior performance and response time of ignoring the size of the congestion window when retransmitting a packet with the fast retransmit procedure. The bursty behaviour of SCTP on recovery from packet loss with larger window sizes can be resolved by adoption of a maxburst parameter. All of these recommendations have been submitted to the IETF as a part of the publication process of our ITC SCTP paper in 2001. The IETF has incorporated them in the current version of the IETF SCTP Implementers Guide [Ste03] where the current author is a named contributor. The SCTP Implementers Guide is being used by the IETF to track SCTP issues and publishes the solutions that will be included in the next version of the SCTP specification. The SCTP Reference Implementation (from release 5.02)

has been modified to incorporate all of our suggested changes.

- **Steady-state SCTP primary paths may be modelled with a steady state TCP model for a link in congestion avoidance mode.**
  Through a number of experiments using both event-based simulation and actual SCTP implementations we have demonstrated the applicability of a family of TCP models to SCTP primary paths. The difference between observed and predicted traffic is well within the order of magnitude difference usually considered acceptable for network dimensioning applications.
- **SCTP paths in slow-start mode may be modelled by approximating slow-start dynamics with a Markov Chain.**
  A series of experiments has shown for a wide range of network conditions and offered network loads that a simple analytic model accurately predicts the throughput achieved. The region of likely operating conditions in terms of observed packet loss rates is very robust in terms of model predictions. This method is less accurate when there is so much data being offered to a path that SCTP is likely to transition into congestion avoidance mode.
- **A model for multi-homed SCTP endpoints has been developed.**
  By observing that under normal conditions only retransmitted traffic is sent on the secondary path, the loss rate and throughput on the primary path can be used to calculate the offered load on the secondary path. When coupled with a suitable model for the primary path (in congestion avoidance) and the secondary path (in slow-start) this observation allows us to predict throughput (and hence network load) on the primary and secondary paths of a multi-homed SCTP network. This model has been verified over a wide range of network conditions both via lab-based experiments and extensive simulation. The model has been shown to produce results with an accuracy of within an order of magnitude in all cases and within 10% for the most likely network conditions for well-engineered SCTP networks, such as signalling networks. It has been demonstrated that although the model is based on steady state assumptions that it is also applicable to a range of quasi-persistent sources.

Chapter 6. Performance Analysis of SCTP/DIN Networks

In the last chapter we developed and verified an end to end model for multi-homed SCTP associations. In this chapter we generalise this model to networks of competing multi-homed SCTP associations. This extended model allows us to analyse the likely performance of networks of DIN entities communicating over multi-homed SCTP associations. Our model provides a basis for dimensioning such a network.

In §6.1 we provide an overview of our network model. In §6.2 we provide a detailed description of our network modelling technique. §6.3 presents the analysis of the model's accuracy for a number of DIN scenarios. Then in §6.4 the application of the model to DIN network planning is described including a derivation of a model for DIN SCP performance on a SCTP transport network. Finally we provide a summary and conclusions in §6.5.

6.1. Overview of the SCTP Network Model

The extended SCTP model is a transport model of a network of competing multi-homed SCTP flows of arbitrary topology. Before presenting an overview of our modelling approach we describe a reference scenario.

6.1.1. Reference Network Scenario

The following network scenario is presented in order to ensure consistency of notation and to provide a visualisation of the types of network configuration addressed by the model. A simple network configuration, which includes all of the properties modelled, is considered here for ease of presentation but in general the quantity of nodes and links can be increased at the cost of increasing the computational complexity of the solution. The network topology may also be varied so a wide range of network configurations can be described. It is important to note that a major determinant of solution complexity is the number of routes modelled through the network rather than simply the number of nodes or links.

The general approach to designing the reference network topology is deliberately very similar to existing SS.7 network configurations. Thus we envisage a fully cross-connected network served by

mated pairs of redundant routers (STPs in SS.7) with all edge nodes connected to the network via two separate routers and links. Even though the topologies are so similar the SS.7 and SCTP/IP networks exhibit major differences in the use of the network due to differences in congestion control mechanisms and the lack of link load balancing in SCTP. The latter difference results in the majority of the traffic crossing the primary network routes and a minority crossing the secondary network routes in contrast to SS.7's more balanced use of network resources. Technically this seems a poor trade-off but if the relative cost of the SCTP/IP infrastructure is low enough then it becomes economically viable. In addition it is possible to design new secondary network topologies that reduce the level of redundancy compared to SS.7 (and our model is capable of supporting them) but these are not considered in the current work. It should be noted that SCTP approximately maps onto the MTP Layer 2 and Layer 3 of the SS.7 architecture.

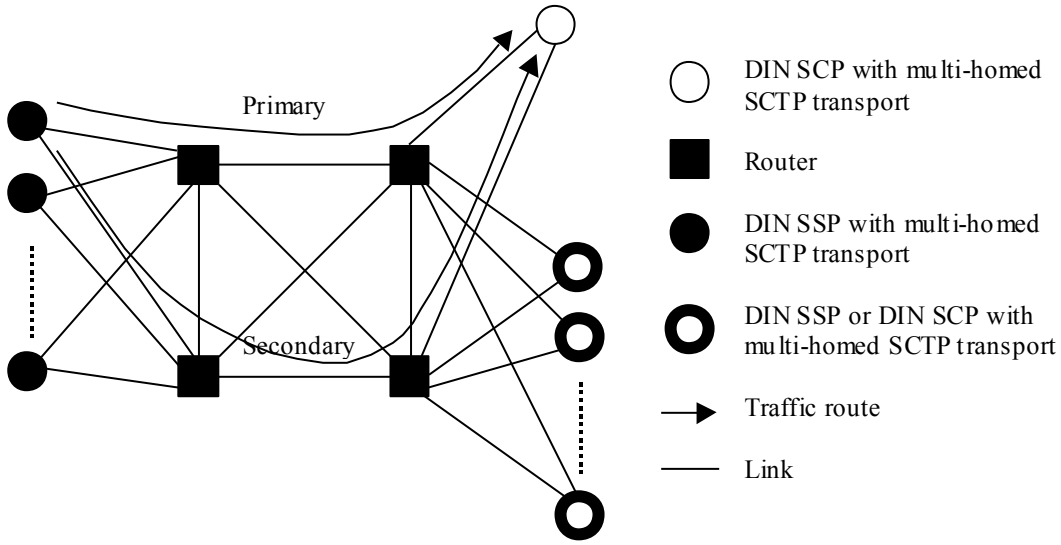


Figure 6.1: Reference Network Scenario

We consider a typical signalling network with a pool of DIN SSP nodes (each one a SCTP multi-homed endpoint) communicating with a DIN SCP node. There is an additional pool of nodes that may represent SSPs or SCPs depending on the precise network scenario examined. The pool of multipurpose nodes would networks take the role of SSPs in traditional signalling networks but a DIN solution could choose to distribute the SCP functionality across multiple nodes. In the analysis that follows we distinguish these cases through the use of different patterns of traffic flows (routes) in the network and thus it has no impact on the formulation of our general analytic model.

In the single-SCP node case, from the perspective of each SSP, the other SSPs form SCTP flows of adaptive traffic competing with its primary SCTP route (figure 6.1). If there are multiple co-operating SCP nodes then there are many possible traffic routing patterns and in each case there will always be a

primary network consisting of competing adaptive flows.

By assuming an engineered network with little cross-over of traffic between the primary and secondary routes we can partition the network into two networks connected only by the multi-homed SCTP endpoints. This is not the approach of the traditional SS.7 network which often load balances traffic between the possible routes however SCTP is incapable of doing this and if there are common bottleneck links on a SCTP primary and secondary path then TCP-style adaptive controls will have a reduced effectiveness when protecting the network. This is because congestion indications on the primary route at the common bottleneck will lead to increased traffic on the secondary route and hence at the congested common bottleneck. Thus compared to the dedicated secondary network case there would be less reduction of congestion on the congested common bottleneck link.

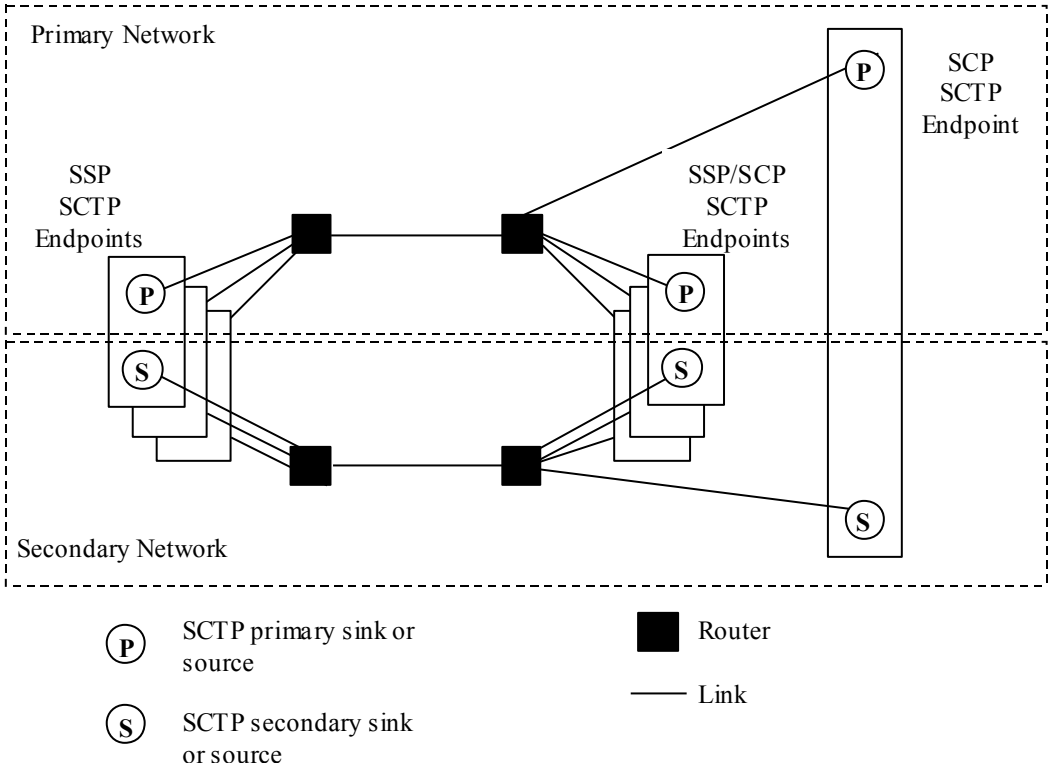


Figure 6.2: Modelled Reference Network Scenario

The network including the SCTP primary routes we often abbreviate as the primary network. The network including the SCTP secondary routes we often abbreviate as the secondary network. We also decompose each SCTP endpoint into two related adaptive traffic sources, one for each network. Each network will be treated separately by a specific sub-model that captures the behaviour of the SCTP endpoint on that network. This approach leads to figure 6.2, our modelled reference network scenario. Arbitrary complexity in terms of routes, links and adaptive traffic source/sink nodes may be added to our primary network. The lightly loaded secondary network is characterised in terms of a single route

between each pair of communicating multi-homed SCTP sink/source nodes.

A number of important quantities for our models can now be defined with respect to the modelled reference network scenario. These are as follows:

$\overline{w}_p$, the average SCTP send window size on the primary route through the primary network.

$\hat{w}_p$, the vector of average window sizes for all routes in the primary network.

p_p, the packet loss probability on the primary route through the primary network.

$\overline{B}_p$, the SCTP primary route average throughput.

$\bar{l}_p$, the average delay or latency on the primary network.

$\overline{w}_s$, the average SCTP send window size on the secondary network.

p_s, the packet loss probability on the secondary route through the secondary network.

$\overline{B}_s$, the SCTP secondary route average throughput.

$\bar{l}_s$, the average delay or latency on the secondary network.

The relationship between flows on the primary network and the amount of data sent on the secondary network is established by a model of the transmission behaviour of the SCTP protocol for multi-homed endpoints. Thus the decomposition of our reference network scenario (fig. 6.1) into our modelled network scenario (fig 6.2) has provided a clear partitioning of the behaviour of the entire system. The behaviour of the entire system can then be predicted by combing the results of four sub-models; one for the queuing behaviour at the DIN nodes, one for a network of flows in congestion avoidance mode, one for a flow in slow-start mode and one for the transmission behaviour of multi-homed SCTP endpoint. The following sections discuss each of these sub-models in turn and finally how the results of the sub-models may be combined to model the behaviour of the entire system.

6.1.2. SCTP Network Model Overview

The SCTP transport model is based upon our observations of SCTP behavior and a new application of standard TCP analytic modelling approximations. By combining our model of a multi-homed SCTP endpoint (see Chapter 5) with an extended version of the TCP network analysis approach of Roughan *et al.* [Rou01] we are able to predict and thus dimension competing flows from an arbitrary topology of multi-homed SCTP endpoints. Roughan's work is discussed in Chapter 2 and an outline of his method is given in §6.2.1.

Briefly, the modelling approach is to partition our network of multi-homed SCTP endpoints into primary (potentially congested) networks of routes and secondary (or redundancy) networks of routes. Our key observation is that under normal operating conditions over long time-scales the maximum safe

operating capacity of the primary routes is a function of the effective throughput on the secondary routes. The following sub-sections give an overview of our primary and secondary network models.

6.1.2.1. Primary Network

The primary network is modelled as a network of heavily loaded routes in congestion avoidance mode. Given a primary network topology and list of desired routes it is possible to estimate the average throughput achieved by numerically solving (via the Newton-Raphson method) a system of non-linear equations of the form:

$$\overline{w}_p = f(p_p(\hat{w}_p)) \qquad (6.1)$$

$$p_p(\hat{w}_p) = g(\hat{w}_p; \hat{c}) \qquad (6.2)$$

Where $\overline{w}_p$ is the average SCTP send window size seen on a single route through the network, $\hat{w}_p$ is the vector of average window sizes for all routes in the network, $p_p(\hat{w}_p)$ is the packet loss probability of a given network route, $\hat{c}$ is a vector of network link capacities and *f* and *g* are functions. The form of *f* is as a vector analogue of (5.1), the "1/*p* law" for TCP networks. The form of *g* is as a vector analogue of Key's [Key98] relationship between observed loss rates and window sizes for competing TCP sources. Both of these expressions are completely defined in the full model description (see §6.2.1). Given these primary route characteristics of average send window size and packet loss probability we can calculate the average throughput and packet delay experienced on the primary routes.

6.1.2.2. Secondary Network

The secondary network is modelled as a network of lightly loaded paths in slow start mode. Our observation of the relationship between the average amount of data SCTP sends on the secondary route ($\overline{d}_s$) and the primary route characteristics of average throughput ($\overline{B}_p$) and packet loss probability (p_p) is:

$$\overline{d}_s = p_p \overline{B}_p \qquad (6.3)$$

The solution of the primary network model provides this information for each route. With this information, the model of SCTP throughput in slow-start mode, a secondary network topology, a list of desired routes and the assumption that some form of active queue management[3] (such as RED) is deployed in the network it is possible to calculate the average window size and observed packet loss probability on the secondary network. This is achieved by numerically solving (via the Newton-

[3] This allows us to assume that even under lightly load conditions the routes will experience some loss.

Raphson method) an analogue of the system of non-linear equations describing the primary network (6.1), (6.2) for the secondary network. These are of the form:

$$\overline{w}_s = h(p_s(\vec{w}_s), \vec{d}_s(p_p)) \qquad (6.4)$$

$$p_s(\vec{w}_s) = i(\vec{w}_s, \vec{c}) \qquad (6.5)$$

Where $\overline{w}_s$ is the average SCTP send window size seen on a single route through the secondary network, $\vec{w}_s$ is the vector of average window sizes for all routes in the network, $p_s(\vec{w}_s)$ is the packet loss probability of a given network route, $\hat{c}$ is a vector of network link capacities and h and i are functions. The form of h is a vector analogue of (5.8), the average window size of SCTP secondary path in slow-start. The form of i is a vector analogue of Key's [Key98] relationship between observed loss rates and window sizes for competing TCP sources modified to take into account of potential resource under-utilisation on the route. Both of these functions are completely defined in the full model description below (see §6.2.2). From this solution we can calculate the average throughput and round trip times for all routes in the secondary network.

6.1.2.3. Combined SCTP Multi-homed Network

From the solutions for the primary and secondary networks we can estimate the overall average delay and throughput experienced by routes on the multi-homed SCTP network. For maximum performance of the combined SCTP network there must be sufficient capacity in the secondary routes to carry the total load placed on it by multi-homed SCTP sources.

By noting that the safe operating thresholds for the primary network are a function of both the demands on the primary network and the available throughput on the secondary we can also solve a variety of network dimensioning problems. This approach is easily scaleable to large networks of arbitrary topology by only considering finite numbers of routes. Hence this model is applicable to a very wide range of problems and not limited to our reference network scenario.

6.2. Modelling Networks of Multi-homed SCTP Sources

In the previous chapter we modelled individual multi-homed SCTP links or routes. In this section we expand our treatment to deal with networks of competing multi-homed SCTP sources. The main new requirements for modelling networks as opposed to routes are as follows:

- It must be established how multiple sources share resources on a given common network link.
- The observed packet loss probability on each common network link must be a function of the utilisation of that link by all sources on the network rather than an input parameter.

- The problem must be formulated in a way that captures the network topology, active routes and individual link characteristics.
- It must be established how the individual link characteristics of links making up a route combine to give the observed route characteristics which SCTP congestion controls operate on.

As in chapter 5, the approach adopted is to partition the network into a primary and secondary network, solve the primary network system first, then solve the secondary network system (partially based on the primary network solution) and finally to combine the results of primary and secondary networks to give an estimate of the net performance observed by applications making use of the multi-homed SCTP association spanning both networks. Our approach for the primary network is based on the work of Roughan *et al.* [Rou01] for TCP networks, which we extend to cover secondary networks and hence multi-homed SCTP networks.

6.2.1. Primary Network

In this section we generalise our previous expressions for a single SCTP primary route's throughput and latency (5.3, 5.4) to the case where there is a network of competing SCTP primary (congestion avoidance mode) flows. We first deal with a single link and then generalise to a network of links in a particular topology with specific active routes.

6.2.1.1. Single Link Case

Given a group of *n* persistent SCTP sources sharing the same link this will lead to resource contention and ultimately packet drops. In this environment the self-balancing aspects of the SCTP congestion control dynamics (invoked via the packet drops) will ensure that on average each source receives an equal share of the mean total throughput on the route in steady state.

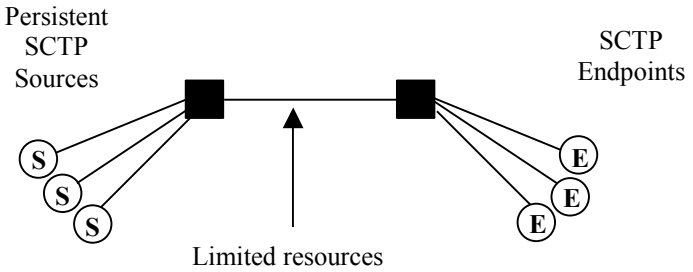


Figure 6.3: Multiple Persistent SCTP Sources Sharing a Single Common Link

Thus for a group of *n* sources sharing the same route that the mean throughput experienced by each source ($\overline{B}_n$) is given by:

$$\overline{B}_n = \frac{1}{n}\overline{B} \qquad (6.6)$$

Where $\overline{B}$ is the mean throughput on the link for the case where there is only a single source. Noting that each source receives the same average throughput and hence will have the same average window size ($\overline{w}$), we can re-write (6.6) in terms of the average window size per source using (5.3):

$$\overline{B} = \frac{\overline{w}nm}{\bar{t}} \qquad (6.7)$$

Where m is the maximum segment size of the link and $\bar{t}$ is the mean round trip time of the link. Note that $\overline{w}$ is a function of p, the observed packet loss rate.

As in chapter 5, another useful property of the route that can now easily be calculated is the mean delay (or latency) experienced by packets traversing the link. Given some distribution of packet sizes with a mean size of $\bar{s}$, the mean delay or latency ($\bar{l}$) experienced by packets traversing the link from a specific source (one of n sources) will be given by:

$$\bar{l} = \frac{n\bar{s}}{\overline{B}} \qquad (6.8)$$

We now have a useful description of the route dynamics (6.7) for multiple sources and a derived system property (6.8). This system description is lacking in one respect, we have described it in terms of the loss rate parameter p whose value is neither specified nor readily derived from network configuration characteristics such as link capacity or delay. In essence how do we determine p for the system without making observations? Following Roughan *et al.* [Rou01] we use the approach of Key [Key98] where p is taken to be a function of $\overline{w}$ and so the relationship between $\overline{w}$ and p (5.2) can be re-written as:

$$\frac{\overline{w}^2}{2}p(\overline{w}) - 1 = 0 \qquad (6.9)$$

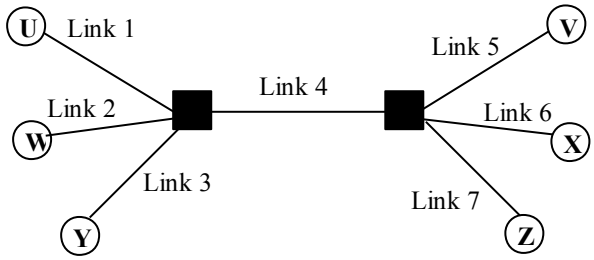
If we can solve, i.e. find the root of, (6.9) then we can determine the system properties of interest, the mean throughput per source and the mean latency experienced by packets traversing the route. However we need an independent relation between our network configuration characteristics (link capacity, delay, offered load) and the packet loss rate to give a solvable system of equations. The assumption made is that for zero buffer capacity at routers the loss rate is the proportional excess of the send rate ($\overline{B}$) over the network link capacity (C) in bits per second i.e.

$$p(\bar{w}) = \frac{\left[\frac{n\bar{w}m}{\bar{t}} - C\right]^{+}}{\frac{n\bar{w}m}{\bar{t}}} \qquad (6.10)$$

Where $[...]^{+}$ takes the positive part. We note that for persistent sources the offered load is greater than the capacity so this is not an issue. We can now combine (6.9) and (6.10) to solve the system for a single link. If desired any of the more complex expressions for the relationship between the window size and loss probability discussed in the literature (see Chapter 2) can be substituted for (5.2) in (6.9) but we retain the simpler approximation of (5.2) here to illustrate the method.

6.2.1.2. Generalising to Networks of Links and Multiple Routes

Roughan *et al.* extend this link model to a network of flows by making the assumption that losses are independent to each other. Hence network route loss rates can be estimated as the sum of the link loss rates on the links which make up the route.



Routes:
U to V, W to X and Y to Z

$$A = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

Figure 6.4: Representing a SCTP Network Topology

To formulate the problem in terms of the network topology and active routes we construct a matrix as illustrated in figure 6.4. Consider a network of N links with capacities described by the vector $\hat{c} = (c_j)$ and link delays described by the vector $\hat{d} = (d_j)$ where $j \in \{1,...,N\}$. We consider R active routes in the network defined by the link subsets $r_i \subset \{1,...,N\}$. These routes are then summarised by the R x N routing matrix $A = [a_{ij}]$:

$$a_{ij} = \begin{cases} 1, & if\ j \in r_i \\ 0, & if\ j \notin r_i \end{cases}$$

We also define route parameters for the MSS, $\hat{m} = (m_i)$, the number of persistent sources sharing the

route i, $\hat{n} = (n_i)$. For a symmetric model where the forward path for all packets is the same as the return path for all acknowledgements then the network is defined by duplex links which halves the value of R. For duplex links with a zero buffer assumption the RTT for each route i, $\hat{t} = (t_i)$ can be calculated as twice the contributing link propagation delays on the route:

$$\hat{t} = 2A\hat{d} \qquad (6.10)$$

If we define a per-route average window size vector ($\hat{w}$) then (6.9) generalises for a single route i to:

$$\frac{\overline{w}_i^2 p_i(\hat{w})}{2} - 1 = 0 \qquad (6.11)$$

And defining a diagonal R x R matrix W of the per-route mean window sizes gives an analogue of (6.9) for the whole system:

$$\frac{W^2 \hat{p}(\hat{w})}{2} - \hat{1} = 0 \qquad (6.12)$$

Where $\hat{1}$ is a vector of ones of length R. From our assumption that (6.9) holds separately for each route. From the independent losses assumption we can calculate route losses as a function of link losses $\hat{q}$ as follows:

$$\hat{p}(\hat{w}) = 1 - e^{A\ln(1-\hat{q})} \qquad (6.13)$$

We know that the link losses must be a function of the route window sizes and hence (from (6.10)) the offered link traffics ($\hat{b}$) and the link capacities ($\hat{c}$). The offered link traffics at each link are a function of our routing matrix (A) and the source node send rates ($\hat{f}$).

$$\hat{b} = A^T \hat{f} \qquad (6.14)$$

Where the elements of $\hat{f}$ are defined by (6.7):

$$f_i = \frac{\overline{w}\, n_i m_i}{t_i} \quad \forall\, i \in \{1,...,R\} \qquad (6.15)$$

Under zero buffer assumptions the loss rate is the proportional excess of the offered link traffic over the link capacities. Thus we have:

$$\hat{q} = \left[1 - \frac{\hat{c}}{\hat{b}}\right]$$

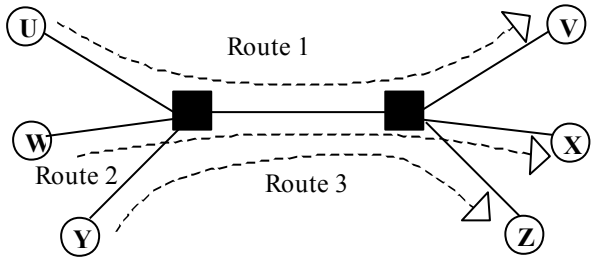
where $\hat{b}$ is the offered link traffics and is given by the expression (6.14).

We now have a system of non-linear vector equations (6.12) and (6.13). Roughan's method is to solve these for a fixed point $\hat{w}^*$ which will give us the steady-state loss probabilities and average window sizes (and hence throughputs and delays) for all routes.

This is equivalent to finding the root of:

$$F(\hat{w}) = \tfrac{W^2}{2}\hat{p}(\hat{w}) - \hat{1} \qquad (6.17)$$

If we take the example network illustrated in figure 6.4 and assume a capacity of 10MBps and delay of 5ms for all routes then the components of the resultant system to be solved are illustrated in figure 6.5.



$$W = \begin{bmatrix} \overline{w}_1^2 & 0 & 0 \\ 0 & \overline{w}_2^2 & 0 \\ 0 & 0 & \overline{w}_3^2 \end{bmatrix}.$$

$$A = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

$$\hat{c}' = \begin{bmatrix} 10e^6 & 10e^6 & 10e^6 & 10e^6 & 10e^6 & 10e^6 & 10e^6 \end{bmatrix}$$

$$\hat{d}' = \begin{bmatrix} 5e^{-3} & 5e^{-3} & 5e^{-3} & 5e^{-3} & 5e^{-3} & 5e^{-3} & 5e^{-3} \end{bmatrix}$$

$$\hat{n}' = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$$

Figure 6.5: Problem Formulation for the Solution of Example Topology

This system can be solved numerically by the Newton-Raphson method [PTVF92]. In practice refinements have to be made to our implementation of the Newton-Raphson method [PTVF92] in order to ensure global convergence of this complex system. But the principal method of making an initial estimate of the solution and iterating to approach the root is the same. For each iteration, $\hat{w}_{n+1} = \hat{w}_n + \delta\hat{w}$, where the $\delta\hat{w}$ are obtained from the equation:

$$J\delta\hat{w} = -F \qquad (6.18)$$

Where *J* is the Jacobian matrix of $F(\hat{w})$. For the system described above, Roughan *et al.* provide the

analytic solution to the Jacobian matrix for the case of the most basic expression of the $1/\sqrt{p}$ law (5.1) which speeds the calculations for this case.

Thus, in terms of our reference network scenario we can find from this model several properties of our combined system. If we define the i^{th} route of the system to be a SCTP multi-homed endpoint's primary source then:

$$\overline{w}_{p,i} = \overline{w}_i \qquad (6.19)$$

, the average SCTP send window size on the i^{th} primary route. Which form the i elements of the vector of primary route window sizes, $\hat{w}_p$. We then define:

$$p_{p,i} = p_i \qquad (6.20)$$

, the packet loss probability on the i^{th} primary route through the primary network. Which form the i elements of the vector of primary route loss rates, $\hat{p}_p$. This allows us to calculate

$$\overline{B}_{p,i} = \frac{\overline{w}_i n_i m_i}{t_i} \qquad (6.21)$$

, the i^{th} SCTP primary route average throughput. Which form the i elements of the vector of primary route throughputs, $\hat{B}_p$.

$$\bar{l}_{p,i} = \frac{\bar{s}_i}{\overline{B}_{p,i}} \qquad (6.22)$$

, the average delay or latency on the primary network (where $\bar{s}_i$ is the mean packet size on the route).

6.2.2. Secondary Network

The treatment of the secondary network is largely as described above for the primary network. A routing matrix and vectors of network parameters are established to describe the network topology and active routes. This is then coupled with expressions governing the dynamic behaviour of the system. To account for the differences between the primary network (in congestion avoidance) and the secondary network (in slow-start) the expressions governing the mean window size and its relation to the packet loss probability, (6.12) and (6.13), are exchanged for the slow-start equivalents described in chapter 5.

This results in the following important modifications:

1. We substitute the slow-start expression for average window size per source (5.8) for the congestion avoidance expression used above (6.12). Hence for a set of sources sharing a single route:

$$\overline{w}_s - \sum_{j=0}^{j\max} 2^j w(2^j) = 0 \qquad (6.23)$$

where,

$$w(2^j) = \frac{s^{2^j-1}}{\sum_{i=0}^{j\max-1} s^{2^i-1} + \frac{s^{2^{j\max}-1}}{1-s^{2^{j\max}}}} \quad \text{and} \quad s = \left(1-p_s(\overline{w}_s)\right)\left(1-\frac{1}{\overline{d}_s}\right)$$

Here we see that the average window on a given secondary route, $\overline{w}_s$, is a function of the window sizes of the other sources transmitting on that route, the slow-start threshold for that route, *jmax*, and the average amount of data sent on that route by the primary source corresponding to each secondary source, $\overline{d}_s$.

Hence, as in the previous section, we have an expression (6.23) of the form:

$$\overline{w}_s - h(\overline{w}_s^z) = 0$$

Where *h* is a function defined in (6.23) and *z* is an exponent related to the size of the slow start threshold configured for these routes.

2. The primary path zero buffers assumption for calculating loss probability will not yield any results for a lightly loaded network as it will often be operating at less than full capacity. Our expression above for average slow-start window size is only defined for positive (but arbitrarily small) loss probabilities. Thus we assume that a combination of the bursty nature of TCP-style traffic and the deployment of some form of active queue management on the network will lead to packet losses. We model this as a small exponential term added to the zero buffer loss rate. The zero buffer term is retained to handle networks where the secondary paths are in fact under-dimensioned compared to the traffic carried. Hence our expression for the loss rate vector, $\overline{q}$, becomes:

$$\hat{q} = \begin{cases} \left[1-\frac{\hat{c}}{\hat{b}}\right] & \text{if } \hat{b} > \hat{c} \\ e^{\hat{b}/\hat{c}} \times 10^{-4} & \text{otherwise} \end{cases} \qquad (6.24)$$

3. We must define a per-route vector ($\hat{d}_s$) for the average amount of data sent by the primary on that secondary route. As in Chapter 5 this can be calculated from the loss rate and throughput on the primary routes (5.10), (6.20), (6.21). Hence:

$$\hat{d}_s = \hat{p}_p \hat{B}_p \qquad (6.25)$$

With these changes in mind we formulate our problem as before in terms of a network of N links and R

active routes summarised by the R x N routing matrix A. We retain the assumption that losses are independent to each other so that network route losses can easily be calculated from link losses. The expression (6.23) above can be expressed in vector form and thus defines a system of non-linear vector equations which can be solved numerically by the Newton-Raphson method. One difference from the primary network solution is the complexity of Jacobian matrix for this system. However the value of this can be estimated numerically via a forward-difference approximation for partial differentiation [PTVF92].

Thus, in terms of our reference network scenario we can find from this model several properties of our combined system:

$\overline{w}_{s,i}$, the mean SCTP send window size on the i^{th} secondary route. Which makes up the components of $\hat{w}_s$, the vector of mean SCTP send window sizes on the secondary route.

$p_{s,i}$, the vector of mean loss rates on the i^{th} secondary route. Which makes up the components of $\hat{p}_s$, the vector of SCTP send window sizes on the secondary route.

$$\overline{B}_{s,i} = \frac{\overline{w}_{s,i} n_i m_i}{t_i} \qquad (6.26)$$

, the mean throughput on the i^{th} SCTP secondary route. Which makes up the components of $\hat{B}_s$, the vector of SCTP secondary route throughputs.

$$\bar{I}_{s,i} = \frac{\bar{s}_i}{\overline{B}_{s,i}} \qquad (6.27)$$

, the average delay or latency on the secondary network on the i^{th} route (where $\bar{s}_i$ is the mean packet size on the route).

6.2.3. Combined SCTP Network

Having solved the primary and secondary network systems of non-linear equations we can estimate the net observed throughput and latency for each SCTP-user application. User data packets are sent on both the primary and secondary routes thus the characteristics of each route and the percentage of data sent on each route will determine the observed characteristics of the combined network. SCTP packet overheads and packets dropped by the network add nothing to the observed performance of the network from a user's point of view so we must account for them in this calculation.

We estimate SCTP packet overheads as follows. Previous work by Jungmaier *et al.* [JST00] has shown

that user data accounts for between 84% and 97% of the data transferred. This depends on the size of user data chunks, whether chunk bundling or fragmentation of PDUs is enabled and the network maximum segment size. We call this value the SCTP user data efficiency factor, *z*, henceforth.

To estimate the net packet drop rate and the share of traffic between the two routes we make the following assumption:

$$p_p >> p_s$$

This follows naturally from our assumption that the primary network is fully loaded and the secondary network is lightly loaded. Thus we can say that approximately p_p percent of the traffic is sent on the secondary network and that there is no loss of this traffic. Hence $(1-p_p)$ percent of the traffic is sent on the primary network.

With these assumptions we can calculate weighted average values for the characteristics of the combined network as observed by an application using the SCTP transport. Thus, for a given SCTP user data efficiency factor, z, on a single multi-homed SCTP association:

$$\overline{B}_i = z\left(p_{p,i}\overline{B}_{s,i} + (1-p_{p,i})\overline{B}_{p,i}\right) \qquad (6.28)$$

is the observed throughput on the i^{th} association.

$$\bar{I}_i = \frac{\bar{s}_i}{\overline{B}_i} \qquad (6.29)$$

is the observed latency or delay on the i^{th} route where $\bar{s}_i$ is the average size of user data messages on the i^{th} route.

Thus we have completed our model of networks of competing SCTP multi-homed associations. This model has two sets of useful results. Firstly there is an estimate of the load placed on the network infrastructure in terms of mean throughputs captured in (6.21) and (6.26) for the primary and secondary networks respectively. Secondly there is an estimate of the aggregate steady-state performance observed by applications using the multi-homed association (6.28) and (6.29). In §6.4 we examine how these latter predictions can be utilised to build stochastic models of application-level performance.

6.3. Analysis of the SCTP Network Model

In the previous sections we developed our model of networks of competing multi-homed SCTP endpoints. In this section we examine the accuracy of our analytic model predictions through comparison with event-based simulation studies of SCTP networks carried out with our OpNet model of SCTP. This evaluation is carried out both for traditional IN-style network topologies and the more

complex topologies that may result from deployment of the DIN. In subsections 6.3.1 and 6.3.2 we describe the network topologies, scenarios and flow patterns modelled. Then in subsection 6.3.3 we present consolidated results for these systems.

6.3.1. Traditional IN Environment Studies

The first set of evaluations of our analytic network model is performed on a group of network topologies that are very similar to the most complex studies available on currently deployed IN/SS.7 networks [Jen01]. As such they represent realistic topologies for existing SS.7 networks carrying IN traffic. Hence we refine our reference network scenario into four test network topologies as illustrated in figure 6.6. Each of these networks features a single multi-homed SCP endpoint, the other endpoints in the network act in the role of multi-homed SSPs. In addition to addressing the case where traditional IN nodes make use of a SCTP transport network, these scenarios may represent the case where DIN SSP nodes provide SSP functionality as a group of collaborating SSF AC objects that are all co-located on a single network element. This may represent either a DIN SSP proxy implementation at an IN to DIN gateway or a native DIN entity. Similarly SCP nodes may represent either a traditional SCP or a group of collaborating SCF AC objects that are all co-located on a single network element. As SCTP, unlike SS.7, does not currently provide for load sharing between network paths on a multi-homed association, the cross-links between the primary and the secondary network are ignored for modelling purposes. A similar simplifying assumption is made when examining SS.7 networks in Jennings [Jen01].

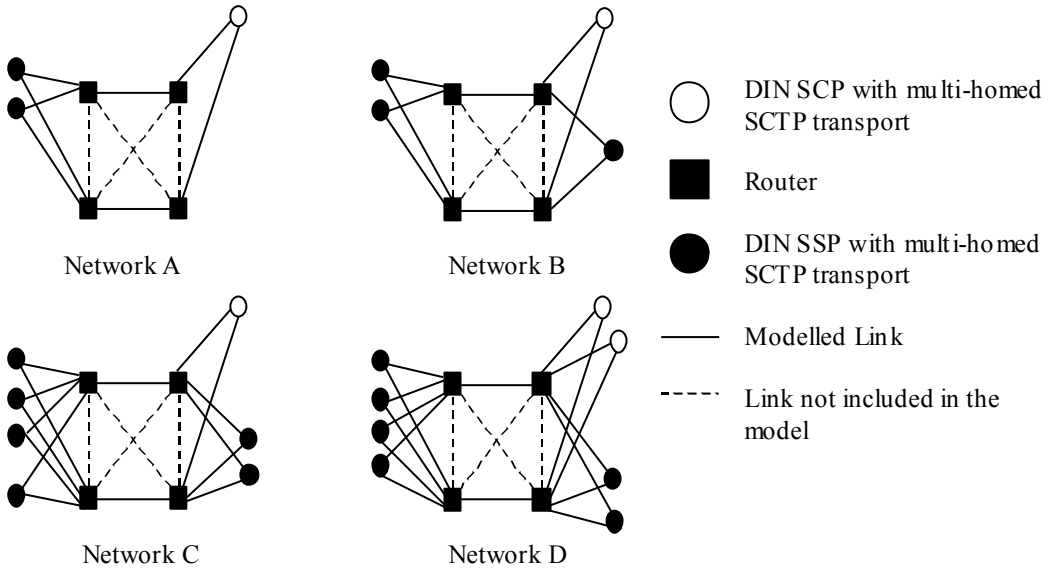


Figure 6.6: Traditional IN-style Test Network Topologies

Each SCTP association modelled represents a full duplex flow of data as service requests are sent to the

other endpoint and responses are returned. Given the ability of SCTP to manage multiple streams within one association without head of line blocking effects [Coe02] it is likely that single long-lived associations will exist between IN entities using a SCTP transport. This also increases the responsiveness and co-ordination of SCTP's reaction to congestion along a single route. This simplifies our modelling as all control traffic between two nodes can therefore be assumed to be on a single association without loss of generality. It should be noted that our model includes the possibility of multiple competing associations along the same route, we just choose to ignore this capability in this scenario for the reasons given above.

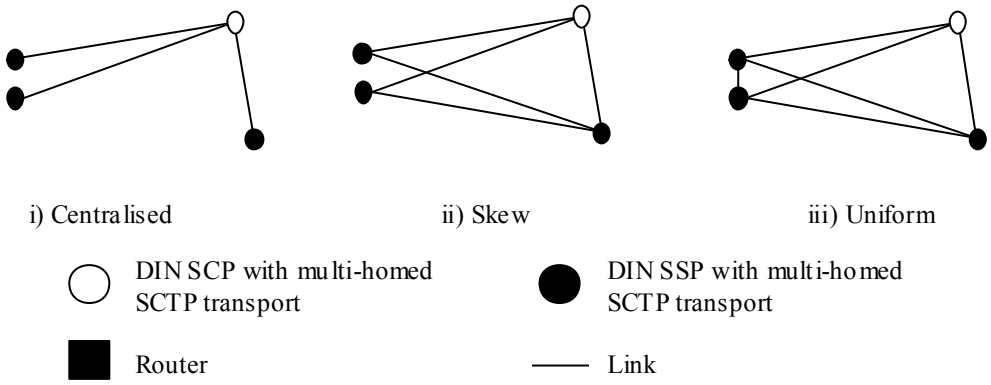


Figure 6.7: Flow Patterns in Traditional IN-style Test Network Topologies

For each network topology modelled we recognise three test flow (association) patterns as shown in figure 6.7 for network topology B. These correspond to the following deployment scenarios:

- Centralised: This is the simplest flow pattern where inter-exchange (ISUP telephony) messages are assumed to provide only a very small fraction of the load on the signalling network. This is either due to a proliferation of service requests such as number portability in a modern network or the presence of either sufficient fully associated links or traditional SS.7 links (not modelled) between SSP nodes to carry ISUP traffic. Hence in this case there is one two-way association modelled for each SSP node (from that SSP to the SCP and back again).
- Skew: In this case we assume that ISUP telephony traffic is present but only occurs between SSPs which are not connected to a common router/STP. All SSPs also route IN service requests to the SCP. This represents the scenario where all SSPs connected to the same router are geographically close and sufficiently connected by associated links that are not modelled. This results in one two-way association for each SSP node to the SCP and one association two-way from each pair of SSPs connected to another router/STP.
- Uniform: This is the most general flow pattern for these types of network topology. In this case ISUP traffic is assumed to flow between all SSPs and each SSP also uses the SCP for IN services. Hence we have one two-way association from each SSP to the SCP and one two-way association

between each pair of SSPs.

In each case these traffic flow patterns would also be valid for SS.7/MTP networks as they represent the tuples of actively collaborating nodes in each scenario.

A comparison of the analytic and simulation model results for these scenarios is included in the "Consolidated Results" sub-section (§6.3.3) below.

6.3.2. DIN Environment Studies

The second set of scenarios for our analytic network model is performed on a group of network topologies that represent either likely DIN deployment scenarios or more complex traditional IN network topologies. A key difference in this environment is that nodes are no longer limited to traditional IN entities. Instead SCP and SSP functionality is split into groups of collaborating objects that are distributed across a number of network nodes. In this work we only consider a basic scenario for the composition and distribution of the DIN entities flowing naturally from our specification translation of ETSI CS-1 as outlined in §4.5.2. It is assumed that each of our network nodes, except for gateways, only contains one type of DIN object. For consideration of more complex DIN deployment scenarios see McArdle *et al.* [McA01] and related work.

The DIN nodes we consider are as follows:

- Gateways: These identical nodes represent a single TC-CORBA gateway with a full compliment of proxy objects performing interaction translation in either the migration or inter-working roles. The most likely application of such gateways is for the interconnection of traditional SSPs with CORBA-based service logic on a DIN SCP so these nodes only make associations to SCP service components.
- SSP Service Components: These nodes implement the interfaces previously defined for SCP_to_SSP_AC_responder, SSP_to_SCP_AC_initiator and assist_handoff_SSP_to_SCP_AC_initiator objects. Hence they make associations only to their corresponding SCP service components (SSP initiator to SCP responder and vice versa).
- SSP ISUP Components: These nodes implement CORBA-based ISUP messaging and communicate only with other SSP ISUP components.
- SCP Service Components: These nodes implement the interfaces previously defined for SCP_to_SSP_AC_initiator, SSP_to_SCP_AC_responder and assist_handoff_SSP_to_SCP_AC_responder objects. Thus they make associations to their corresponding SSP service components (SCP initiator to SSP responder and vice versa) and to their SCP service logic components.
- SCP Service Logic Components: These nodes may represent SDPs or other aspects of service logic that have been distributed from the SCP service components themselves. They only communicate

with their SCP Service Components.

Hence we refine and extend our reference network scenario into two test network topologies as illustrated in figure 6.8. Each of these networks except the last features a single DIN SCP and a variable number of DIN SSPs or gateways. The cross-links between the primary and the secondary network are ignored for modelling purposes.

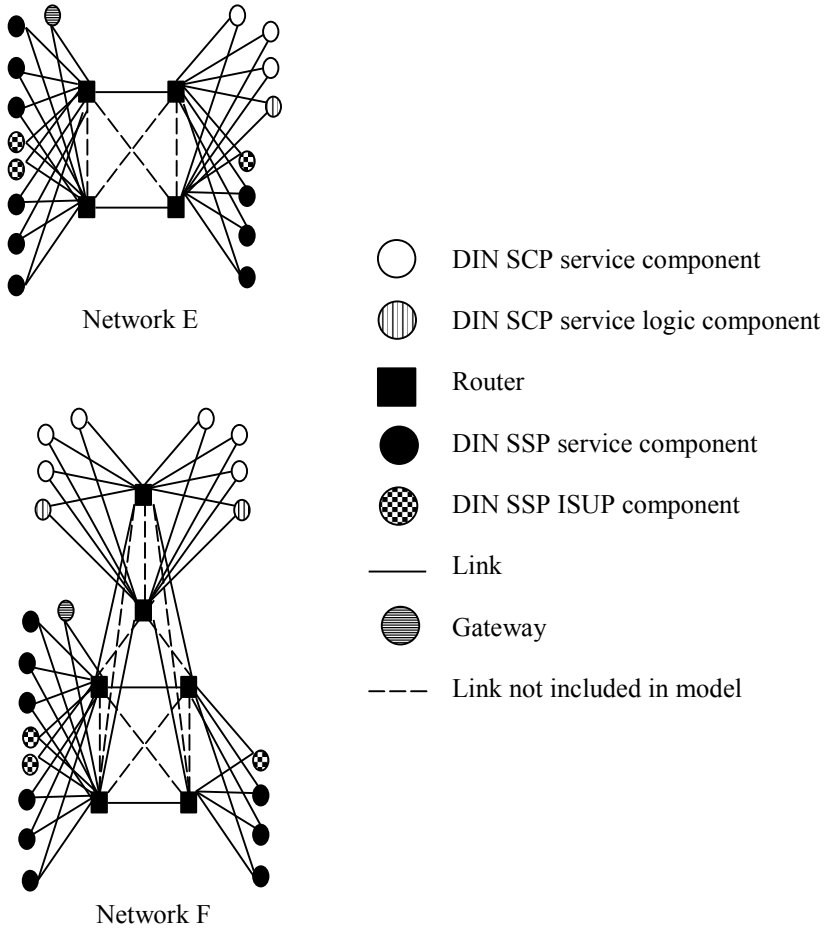


Figure 6.8: DIN Test Network Topologies

The proliferation of node types establishes more complex patterns of traffic flow due to the peer-to-peer nature of many interactions. In terms of our transport network model this means that most DIN nodes (whether gateways, SSP components or SCP components) initiate dialogs with only limited subsets of the other nodes. For each network topology modelled we recognise four test flow (association) patterns as shown in figure 6.8 for network topology E. These are as follows:

- Centralised: This is the same as the centralised flow pattern described above for traditional networks. However associations are further constrained to only take place between application context pairs (initiators and responders).

- Skew: This is the same as the skew flow pattern described above for traditional networks. However associations are further constrained to only take place between application context pairs (initiators and responders).
- Uniform – 50%: Rather than modelling every possible route in the network, as was done in the case of traditional IN scenarios, instead each node has a 50% chance of creating an association to every other node of an eligible type. In this case the greater diversity of network nodes has produced some optimal routing patterns for load balancing, fault tolerance or other purposes. This is in effect a very simple simulation of the possibilities for such controls to be applied. See McArdle *et al.* [McA01] and related work for further discussion of load balancing and optimal object placement in the DIN.

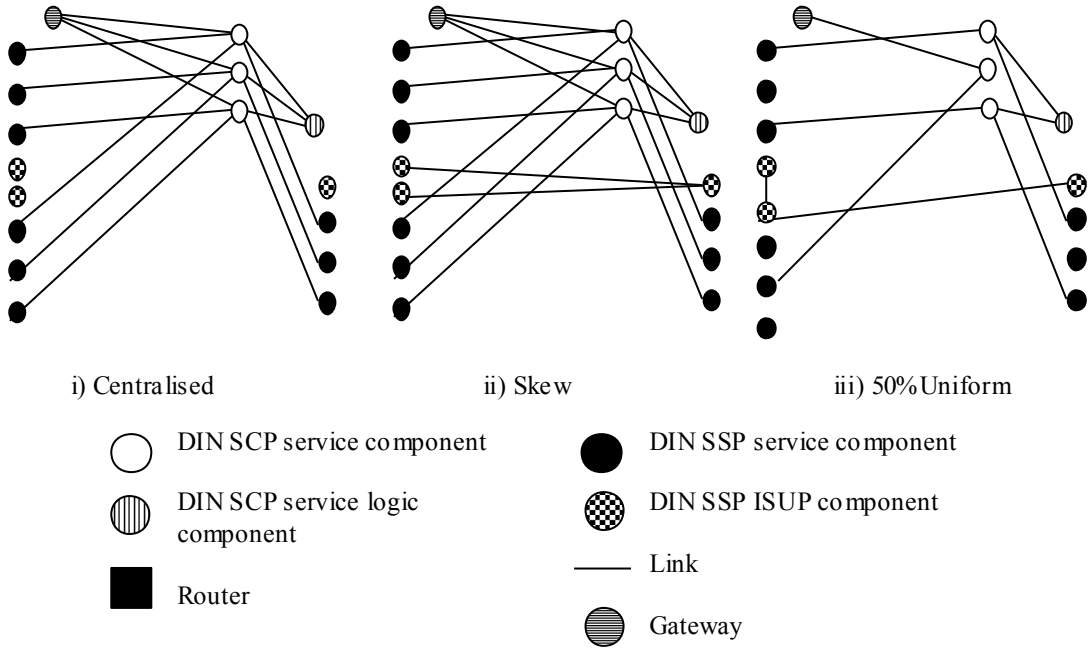


Figure 6.9: Flow Patterns in DIN Test Network Topologies

For each network topology and flow pattern the traffic characteristics predicted by the analytic model and simulated under our OpNet model are then compared under a wide variety of network parameter values as in the last section. The results of these experiments are summarised in the following three subsections. Each of these sub-sections analyses the results in terms of the congestion windows for both the primary and secondary routes, the aggregate throughput of an association and the average latency experienced by individual data packets.

A comparison of the analytic and simulation model results for these scenarios is included in the "Consolidated Results" sub-section (§6.3.3) below.

6.3.3. Consolidated Results

In this sub-section we compare the traffic characteristics predicted by our analytic model and simulated under our OpNet model for each network topology and flow pattern described in the previous sub-sections.

Each network scenario was investigated under a wide variety of network parameter values for the link capacity, link delay, maximum segment size and queuing discipline at the routers (droptail or RED). The simulation used four typical values for MTU: 576, 1500, 4352 and 9192 bytes. The one-way per link delay spanned values between 3ms and 30ms. The link bandwidths ranged from 100kb/s to 10Mb/s. ThePacket loss probabilities were generated from $1e^{-4}$ to $1e^{-1}$. Router buffers were configured according to the bandwidth delay product rule with a ratio of between 15:1 and 1:400.

In each case we simulated the transfer of several thousand unique data packets across each route in OpNet with normally distributed association start times. This ensured that time was allowed for the system to settle into equilibrium with over fifty congestion avoidance cycles (between loss events) taking place in each primary route. The results of these experiments are summarised in the following figures.

We examine the predictions of the system in terms of three model characteristics, the mean aggregate throughput of an association ($\overline{B}$), the mean latency experienced by individual data packets ($\bar{I}$) and the congestion windows for both the primary and secondary routes ($\overline{w}_p, \overline{w}_s$). Given accurate models, these characteristics can be used to make network-planning decisions about a transport network infrastructure. This information can also be used for the planning and configuration of DIN nodes on such a transport network (see §6.4).

There are two types of plots shown, the first type examines the accuracy of the model predictions by plotting them against measurements obtained from event-based simulation results for the same network scenario. The second type of plot presented shows a histogram of the relative error in the model's predictions for each network property already plotted. A dashed line at 45 degrees on the first type of plot indicates the theoretical ideal result of a perfect match between simulation and model.

Before examining these results in more detail we must stress that our method is based on techniques that are well suited to providing a rapid, "zeroth order" approximation for system performance rather than a precise but intractable problem formulation. Another important feature of our work is the choice of realistic but difficult to solve homogeneous link signalling network scenarios. Most work utilising fixed-point approximations [Gib00, BT01, Fir01, Rou01] instead focuses on extremely asymmetric network scenarios or even single congested router/link approximations for the networks. These systems have pronounced, even intuitive, fixed points about which the system tends to rapidly stabilise. This is often not the case for our signalling network topologies where more symmetric demands and link properties make convergence more difficult. However the robustness of these methods ensures that

convergence does occur in all of the scenarios tested and the results are still useful for a variety of applications.

6.3.3.1. Aggregate Results for Each Association

In this section we present plots of the aggregate characteristics (throughput and latency) experienced on each multi-homed association based on the model predictions outlined in §6.2.3. The aggregate throughput of an association represents the mean throughput observed by an application using a given multi-homed SCTP association. The aggregate latency represents the mean transfer time for a data packet of the maximum IP datagram size for the route. We build on these results in §6.4 to estimate the performance of DIN SCP nodes linked by such a multi-homed SCTP transport network.

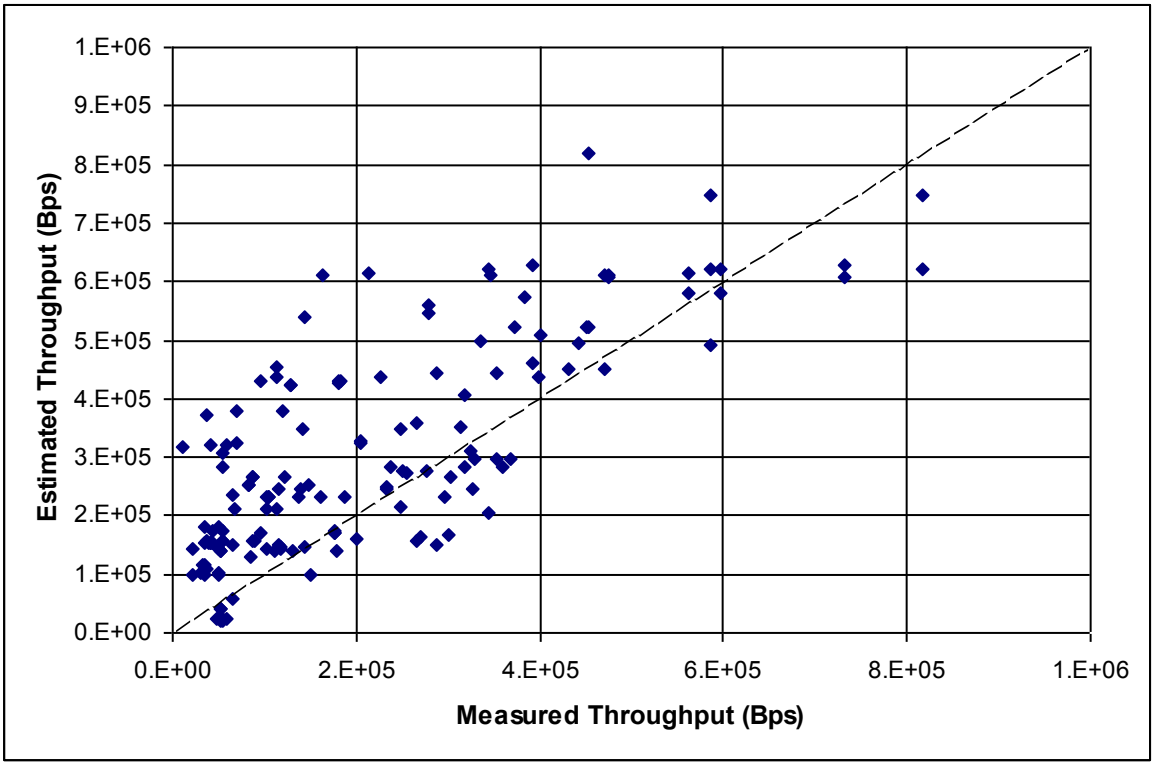


Figure 6.10: Comparison of Model and Simulation Results for Association Aggregate Throughput

Figure 6.10 shows a plot of the mean throughput estimated by our fixed-point method and the actual throughput measured in simulations of the same network scenario. In general the data is a good fit with a bias towards over-estimating the throughput achieved. There is a significant amount of spread in the results achieved. This is explained by the lack of any consideration of queuing effects within the network itself. The level of relative error tends to increase for lower throughput flows which represent more complex network scenarios and ones for which queuing effects are more dominant on transfer times. Corrections to incorporate a more sophisticated model of network delays and the implicit inefficiencies in TCP-style flow controls are possible but are not considered here. The vast majority of

flows exhibit a level of error that is within one order of magnitude of the estimated value. At the upper end of the throughputs illustrated there seems to be a stronger tendency to under-estimate the actual throughput achieved but this observation must be tempered with the knowledge that there is relatively few data points in this region of the figure.

In figure 6.11 the model's estimate of mean aggregate latency is plotted against the simulation results. This is a log scale plot that shows a good estimate of the trend in latency values for the various network scenarios but there is considerable deviation between the estimated and observed values. The impact of extensive queuing in the network can readily be observed in the distribution of points below the line showing the ideal match between model estimates and simulation measurements. In general there is agreement within an order of magnitude between the estimated and measured values however for networks with large queue lengths this can easily be exceeded. This tendency to underestimate transfer delays is combined with a tendency to underestimate congestion windows (see next subsection) to produce the approximately correct predictions for throughput observed in the last figure.

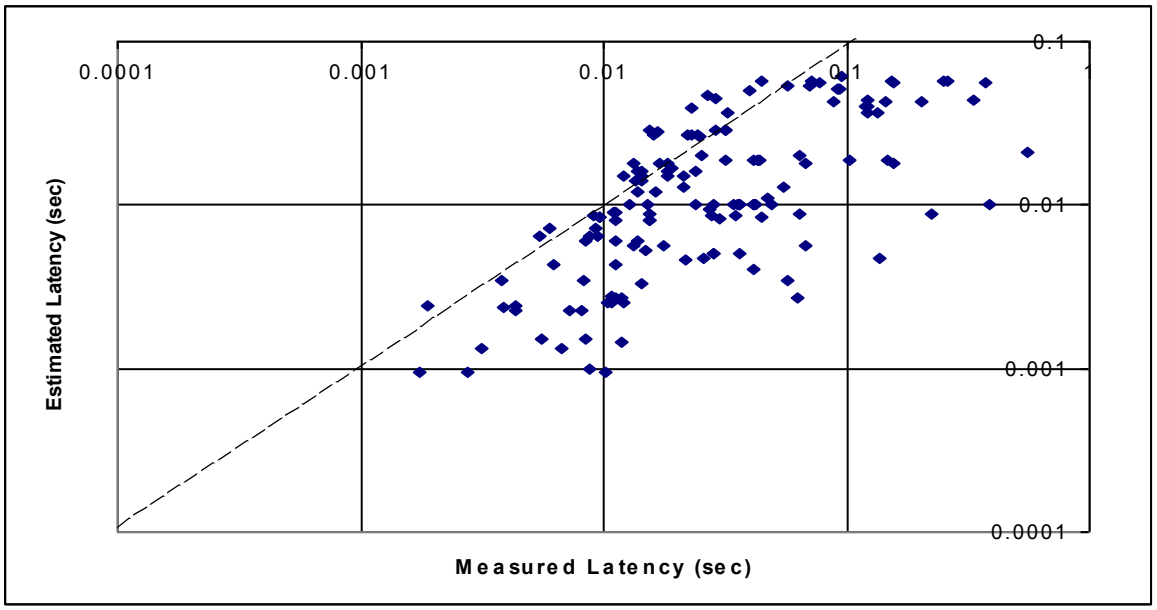


Figure 6.11: Comparison of Model and Simulation Results for Association Mean Latency

In the figure below (6.12) we illustrate the histograms of relative error for the mean aggregate throughput and latency. These show a desirable distribution of relative errors with the majority of all predictions being for lower values of error. For throughput the vast majority of estimates are correct within +/- 100% and for latency the vast majority of estimates are correct within an order of magnitude. As the relative error increases the number of instances observed decreases rapidly. For all of the scenarios examined the throughput relative error was less than an order of magnitude. This is a good result and we can say with confidence that our method produces high quality estimates for the throughput observed on a given route. The latency predictions are less robust but of course these exhibit

a greater dependency on queuing effects within the network.

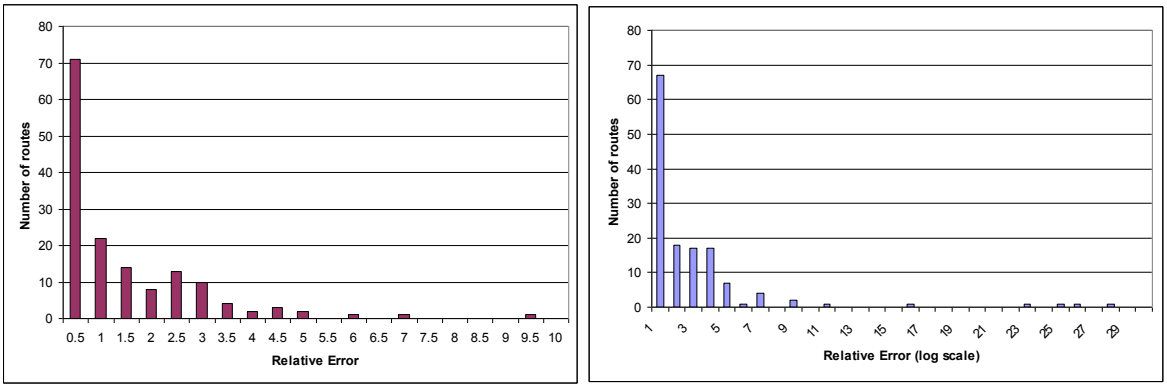


Figure 6.12: Histogram of Relative Error of the Results for Association Aggregate Throughput (left) and Latency (right)

6.3.3.2. Primary and Secondary Path Results

The measure of accuracy examined in this section is the analytic model predictions for primary and secondary congestion windows. The dynamic system examined by our model determines the distribution of congestion window sizes for a given network scenario. This determines the proportion of available network resources allocated to each route via the interaction of the SCTP congestion control and router queue management mechanisms.

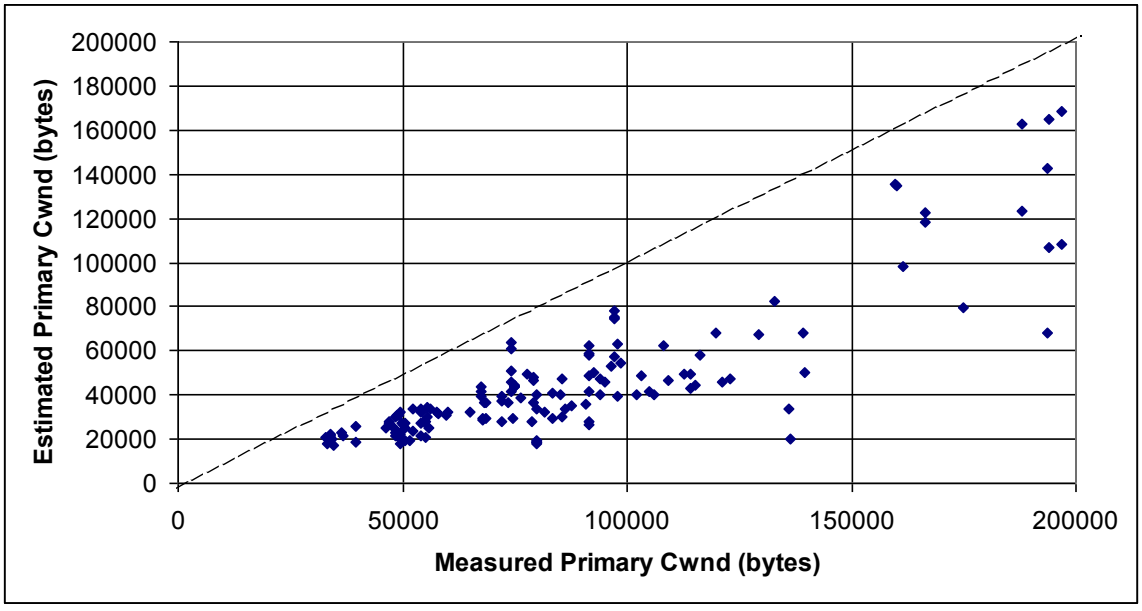


Figure 6.13: Comparison of Model and Simulation Results for the Primary path

Figure 6.13 illustrates the results for the value of the congestion window on the primary path. In all cases the model predicts a smaller window value than is actually achieved. This is consistent with

earlier results for this method as applied to TCP flows [Rou01]. However the general trend of the results is correct and the error (see fig. 6.15) is acceptable given the relative simplicity (and hence ease of computation) of the model. See §6.3.3.4 for more information on the computational costs.

The wedge-shape of the plot also suggests that the distribution of error is relatively uniform across the wide range of predicted congestion window values and network conditions tested. There are some outlying points, such as in the region of a measured congestion window of 140,000 but these can be attributed to simulated systems that have not really reached equilibrium despite our best efforts when simulating. This can be established by looking at the relative congestion windows for similar flows within the same simulation run. The most important aspect of this plot is the illustration of the method's ability to relatively accurately distribute the proportions of network resources to individual routes, even if the absolute values of the congestion windows are under-estimated. This under-estimation is more pronounced for scenarios where queuing time in the network is a significant proportion of the total packet transfer time. Heuristic corrections for this feature of the method are suggested in [Rou01] but they are not pursued here. We instead present the base accuracy of the general method for multi-homed routes. Future work can address refining these results.

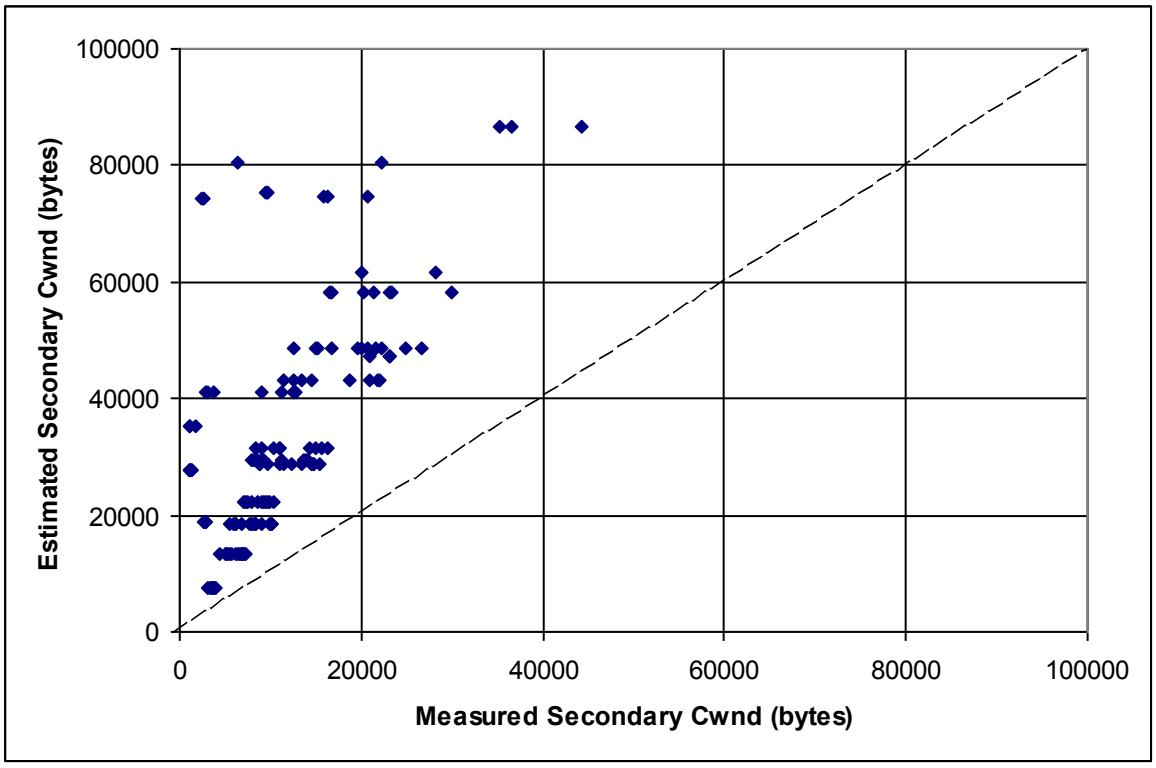


Figure 6.14: Comparison of Model and Simulation Results for the Secondary path

Figure 6.14 illustrates the congestion window results for the secondary network. It is noteworthy that the margin of error in the secondary results is generally larger than that for the primary network. This is

to be expected when dealing with flows with much smaller numbers of packets and hence a larger degree of variability in the simulation results. This could be adjusted for by simulating the transfer of larger amounts of data but since many of the data points on these plots each represent over 10 hours of simulation time on a modern desktop PC there is a practical limit to the amount of realistic secondary traffic that can be simulated. See figure 6.15 for a plot of the relative error between the model and simulations. However once again we have a generally wedge-shaped plot implying a relatively constant spread of the relative error. One feature of the plot is a relatively large number of outlying data points for very small windows, indicating that the model can be very inaccurate in this region. These results are generally for simulation runs where large router queues in the primary network ensure much lower packet drop rates than our model predicts. This has a knock-on effect in our secondary network model where there is correspondingly little data to transmit and hence little potential for congestion window growth.

The next figure, 6.15, shows the distribution of the relative error of the congestion window results for both the primary (left) and secondary (right) paths. We define the relative error as used here as the difference between the simulated and predicted results divided by the simulation results. The causes and distribution of error have already been discussed above but at this stage it is worth noting that whilst our secondary route model is much less accurate than the primary route model that over 95% of the model's estimates are within an order of magnitude of the values measured in the simulations. This is a reasonable level of accuracy for some network planning tasks.

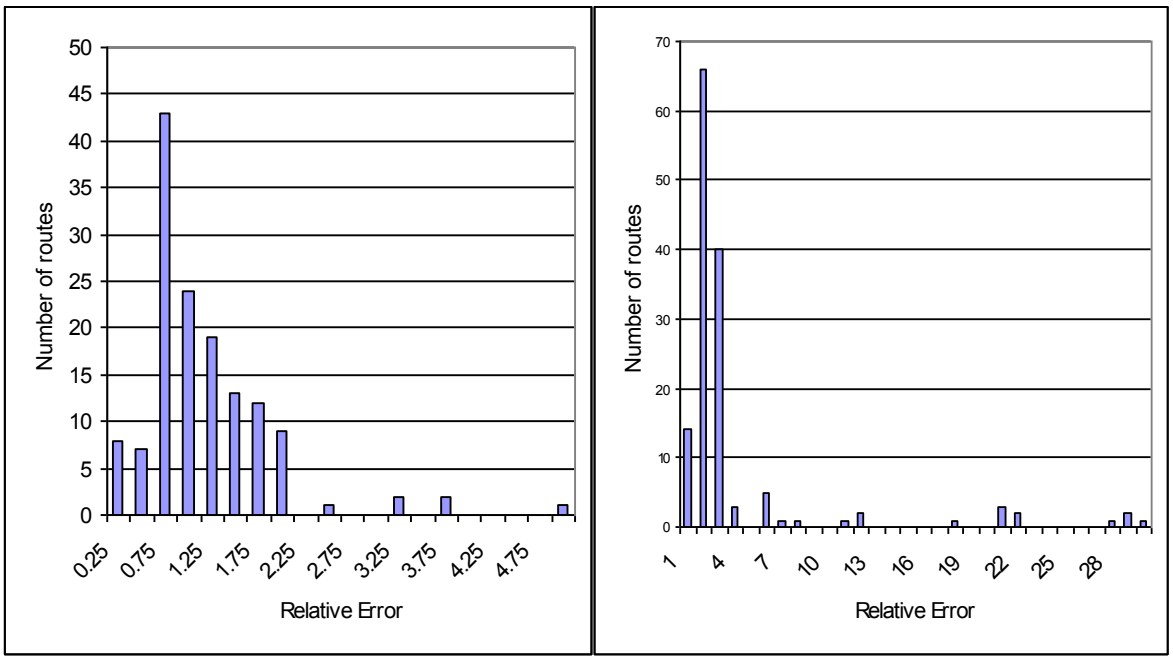


Figure 6.15: Histograms of Relative Error for the primary (left) and secondary (right) networks

6.3.3.3. Computational Costs

In addition to providing insight into the dynamics of a system a major reason for the formulation of an analytic model is the relative ease with which it allows the effects of changing model parameter values to be calculated. In this section we present a comparison of the relative processing costs for evaluating our chosen signalling network topologies with our event based simulation model and the fixed point approximation of the analytic model. Both of these evaluations were carried out on the same standard desktop PC. The Opnet simulation environment was used for the event-based simulations and our fixed-point method was implemented a set of collaborating MATLAB scripts.

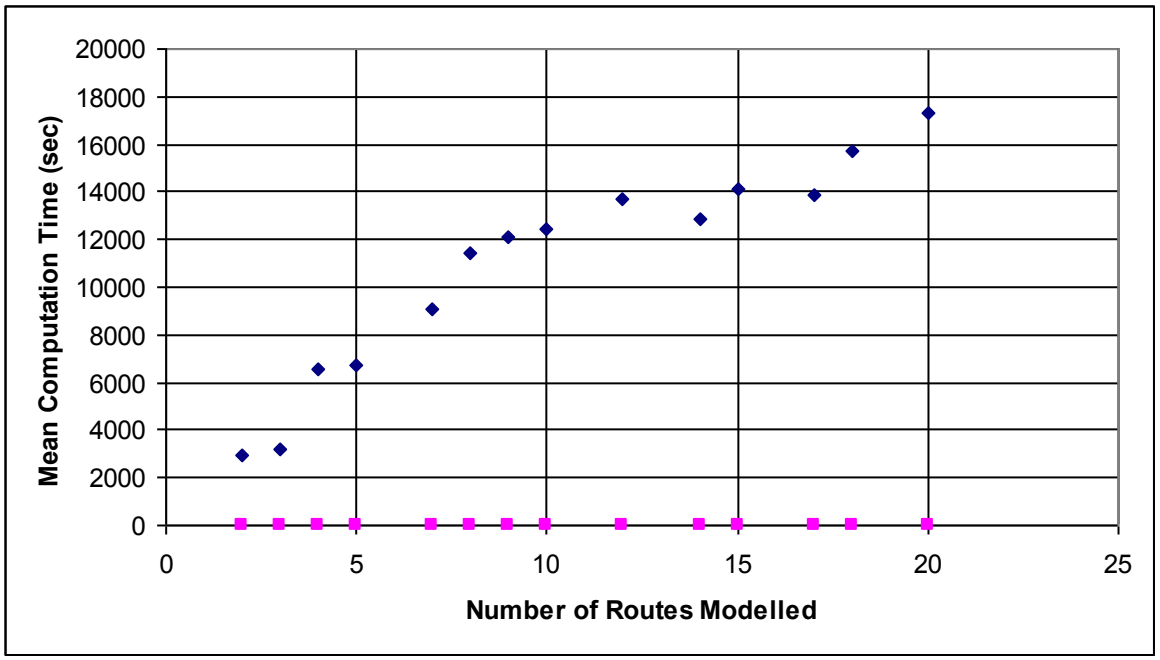


Figure 6.16: Comparison of Mean Computation Time for Fixed Point (lower points) Method and Event-based Simulation (upper points)

We can see from figure 6.16 that the fixed point method is much more efficient and rapidly increases in utility as the number of network routes grows.

In all cases the fixed point method converged rapidly (within 100 iterations) and produced a solution within a second compared to the event-based simulation results, which took from minutes to several hours to execute (depending on the complexity of the scenario). Although 0.66 seconds was the largest execution time recorded for the fixed point method it must be noted that no approximately linear growth in computation time with number of routes was observed for this method. Instead the complexity of the problem in terms of the relative ease of convergence of the system dominated. Thus a slowly converging system with 10 routes could take longer to calculate than a rapidly converging system with 20 routes. An example of rapid convergence would be a system with a single point of congestion and widely asymmetric loads and link capacities. Well-engineered signalling networks, like

our example network scenarios, tend towards slow convergence. Despite this all of the systems converged rapidly from either uniform initial conditions (for the simpler scenarios) or very crude guesses of likely bottlenecks set at half the uniform, rate.

One computational cost that is not included in the plot above is of course the development, debugging and maintenance time involved in creating the software for each model. The event-based simulation code is approximately 15,000 lines of C++, which is dependent on the large and complex application-programming interface (support libraries) provided by OpNet. Whereas the MATLAB scripts for the fixed point method represent less than 300 lines of code which is dependent on the much simpler environment of standard function calls provided by MATLAB.

6.4. Application of the Transport Model to DIN Network Planning

In the previous section we have shown how our analytic model for networks of competing multi-homed SCTP flows can be used to predict network performance for a large variety of network topologies and active flow patterns. Most significantly the relatively simple relations utilised in the model allow us to rapidly solve the system of equations on a standard desktop PC. Thus a wide range of network planning or dimensioning tasks can be efficiently tackled using our approach. In this section we discuss possible applications of our model to planning for the SCTP/DIN environment. First we outline a methodology for estimating the performance of DIN nodes deployed in such a network and then more briefly we discuss applications for SCTP network configuration and dimensioning studies.

6.4.1. Estimating the Performance of DIN Nodes

In this section we use our model of the SCTP transport network, a standard Bellcore model of IN query time and derived characteristics of the DIN and ORB specifications to estimate the likely performance of DIN network elements communicating over multi-homed SCTP networks.

In this treatment we assume a traditional partitioning of IN functionality into SCP and SSP nodes. Thus our DIN is directly comparable with the IN functional architecture [ITU97] where CORBA over SCTP has been used to replace the SS.7 infrastructure. This follows from our focus on transport issues. However this treatment naturally generalises to many of the less complex DIN deployment scenarios where service logic objects and application context objects are deployed on the same network element. For exploration of the performance implications of the more flexible deployment configurations possible with a DIN architecture see [McA01] and related work.

6.4.1.1. Methodology

Bellcore [Bell97] have defined a standard model for estimating IN SSP-SCP query response times. This is illustrated in figure 6.17 below. When used in conjunction with standard benchmark IN applications

this model can be used to define requirements for IN platforms such as SCPs. The reference mean values for the overall query time and various components, such as disk access time, of the query time are published by both Bellcore and the ITU.

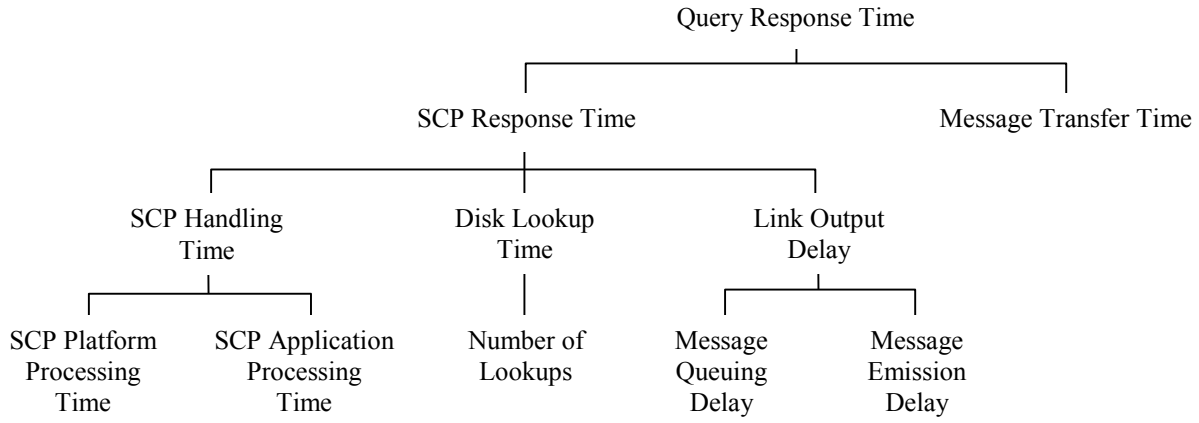


Figure 6.17: Bellcore Standard SCP Query Response Time Model

When we examine this model as applied to DIN networks we can see that two key differences are the calculation of the message transfer time (based on SCTP/IP network rather than SS.7), Link output delay (based on SCTP rather than SS.7) and the precise definition of SCP Handling time (CORBA IDL/GIOP rather than TCAP/INAP effects must be included here).

In the Bellcore standard, the SCP Handling Time is defined as the interval that begins when the last bit of a Call Related message enters the SCP, and ends when the last bit of a Call Related message is placed at the outgoing signalling link buffer, excluding time taken for a disk lookup. It is further subdivided into the SCP Platform Processing Time and the SCP Application Processing Time. SCP Application Processing Time is the application-specific processing time for a particular IN service implementation, it is assumed to be unaffected by the DIN changes to the platform. SCP Platform Processing Time is broken down into three components: TCAP/INAP decoding time, IN platform support processing time and TCAP/INAP encoding time. We will consider the IN platform support processing time to be unaffected by our use of the DIN. TCAP/INAP decoding and encoding times must be modified in order to account for the overheads introduced by using CORBA-based messages that rely on IDL specifications and GIOP marshalling (encoding) and de-marshalling (decoding).

The calculation of the Link Output Delay must also be modified to take into account typical DIN message sizes instead of TCAP/INAP message sizes and also the performance characteristics of networks of competing multi-homed SCTP flows.

Hence we can express the mean total query response time, t_q, for a service as follows:

$$t_q = t_p + t_a + nt_d + t_o + t_t \quad (6.33)$$

Where t_p is the mean SCP platform processing time, t_a is the mean SCP application processing time, n is the number of disk lookups, t_d is the mean disk lookup time, t_o is the mean link output delay and t_t is the mean message transfer time.

6.4.1.2. ORB Processing Costs

Experimental evidence [MBJVC00] has shown that the average TCAP/INAP message encoding/decoding time for a reference platform is on the order of 1ms and that INAP GIOP/IDL message marshalling/de-marshalling time is approximately 50% greater. However when we compare this time with the mean of 100ms allocated to SCP Platform Processing time in the Bellcore model it is clear that any delays due to GIOP/IDL will be minor when compared with overall platform processing time. In the interest of caution, a safe assumption might be to take the peak expected TCAP/INAP message decoding/encoding time of 10ms and allow for an additional 5ms overhead when using GIOP/IDL. As the SCP Platform Processing time includes both an encode and a decode operation, by using GIOP/IDL we will add 10ms to the standard mean SCP Platform Processing time. A Normal distribution of processing times is assumed as in the Bellcore model.

Another significant impact on overall SCP response time will be the larger message sizes generated by GIOP/IDL encoding instead of TCAP/INAP/ASN.1. Bellcore use 279 octets as a typical value for TCAP/INAP encoded with ASN.1 basic encoding rules (BER). Even for traditional IN systems it is likely that this value is now obsolete due to the increasing complexity of signalling messages within the IN [Jen01]. However if we assume that this value represents a typical simple IN query then we may make a comparison by examining the likely size of an equivalent query when encoded as DIN IDL/GIOP. Due to the inefficiencies of the JIDM ASN.1 to IDL mapping and the complex types generated, we assume that an IDL/GIOP message encoded with the OMG's Common Data Representation (CDR) encoding rules will be at least twice the size of the equivalent INAP/BER message. Thus we use a value of 558 octets as our mean message size.

6.4.1.3. SCTP Link Output Delay

The final element of the SCP response time to be calculated is the link output delay which is made up of the queuing delay and emission delay. Queuing delay is a function of link occupancy and the message length distribution. Emission delay is a function of the signaling link speed and the message length distribution. The Bellcore approach is to treat the local link and buffer as an M/G/1 queue and use Pollaczek-Khintchine's formula [ITUTele] with a Palm form factor of 4 for the message length distribution at a load of 0.4 Erlang to calculate the mean waiting time in this system. For this calculation the average message size in octets divided by the link speed is used to calculate the mean

service time. This accounts for both the Queuing delay and the Emission delay components of the Link Output Delay for a given load on the SS.7 network. A value of 0.4 Erlang is picked for an individual SS.7 link to provide a safe operating threshold of 0.8 Erlang if one of the paired load sharing links fails.

For the SCTP case we modify this method as follows. We retain the assumption of Poisson arrivals for data to be transmitted on the link (human –initiated service requests) and will continue to treat the combined system of the SCTP output buffer and link as a M/G/1 queue. Where we differ is in our derivation of the service time for the system. Characterising a SCTP/IP or TCP/IP network using a simple queuing system is problematic due to the closed loop nature of SCTP or TCP congestion control [Arv99, Arv01]. The fact that there may be a 100MB Ethernet connection between our nodes is irrelevant if SCTP congestion control mechanisms will not allow us to send more data and queues the data locally. However if we apply our network model for multi-homed SCTP endpoints then we can estimate the average effective throughput of the association. In chapter 5 we demonstrated the wide range of near peak rates for which this assumption is valid for routes in congestion avoidance. Hence we calculate the mean service rate in terms of the average effective throughput of the multi-homed association. Similarly loading on the link is calculated in terms of average effective throughput rather than theoretical link speed. Hence our link output delay is given by Pollaczek-Khintchine's standard result for mean waiting time in a M/G/1 queue (3.28) as:

$$t_o = \frac{A m_{GIOP}}{2(1-A)\overline{B}}\varepsilon \qquad (6.34)$$

Where A is the offered load in terms of the fraction of the effective transmission rate from our transport network model, $\overline{B}$ is the average throughput on the route calculated by our model, m_{GIOP} is the average message size of INAP packets encoded as GIOP/CDR and ε is the Palm form factor of the distribution of service times (*c.f.* §3.2.1.5).

The form factor for INAP over GIOP/SCTP traffic will be based on the distribution of message sizes from the specification translation of INAP/BER messages into GIOP/CDR. If we assume, as above, that this results in a doubling of message size and hence service time then we can calculate the new form factor from the SS.7 case as follows. Bellcore assume a value of 4 for the form factor in the SS.7 case. The expression for the form factor (3.29) is:

$$\varepsilon_{ss7} = 1 + \left(\frac{\sigma_{ss7}}{s_{ss7}}\right)^2 = 4 \qquad (6.35)$$

Hence

$$\left(\frac{\sigma_{ss7}}{s_{ss7}}\right) = \sqrt{3} \qquad (6.36)$$

From our assumptions, $\sigma_{sctp} = 4\sigma_{ss7}$, $s_{sctp} = 2s_{ss7}$, so

$$\left(\frac{\sigma_{sctp}}{s_{sctp}}\right) = \left(\frac{4\sigma_{ss7}}{2s_{ss7}}\right) = 2\left(\frac{\sigma_{ss7}}{s_{ss7}}\right) \qquad (6.37)$$

Therefore we can calculate ε_{sctp} from (3.29), (6.36) and (6.37) as:

$$\varepsilon_{sctp} = 1 + \left(2\sqrt{3}\right)^2 = 13 \qquad (6.38)$$

Thus (6.34) and (6.38) define our link output delay plus transfer delay in a CORBA over SCTP DIN SCP.

6.4.1.4. Standard Performance Requirements

Bellcore specify [Bell97] industry standards for SCP response times for a simple TCAP transaction (one query, one response, for data retrieval from a LIDB (Line Information DataBase) for an operator system). These are summarised in table 6.1.

	Reference Load A (Daily Peak)	Reference Load B (Yearly Peak)
Mean Value	<= 250ms	<= 400ms
0.95 probability will not exceed	300ms	600ms

Table 6.1: Bellcore Recommended Values for Query Response Times

Bellcore [Bell97] state that the standard SCP Application Processing Time is 25ms and that a single disk lookup is required for this application.

Given the calculations above and the solution to our transport network model for a given dimensioning scenario it is possible to evaluate the likely performance of DIN SCP nodes in the modelled network. From (6.33) and our discussion above we have the following parameter values:

The mean platform processing time, $t_p = 110ms$

The mean application processing time, $t_a = 25ms$

The mean disk lookup time, $t_d = 30ms$

The number of disk lookups, $n = 1$

Thus we can now re-state (6.33) for the purposes of this standard test as:

$$t_q = 165ms + t_o + t_t \qquad (6.39)$$

Where $t_t + t_o$ are calculated from the queuing model of a CORBA/SCTP DIN SCP (6.34) and (6.38).

Hence given the model's solution to a given network scenario and the calculations above we can now determine if the system can meet Bellcore's performance requirements for the total query response time.

6.4.1.5. Experimental Evaluation

In this section we use the method outlined in the previous section to predict the performance of SCP nodes in our reference network scenarios for a variety of offered loads. We compare the predicted delays with the observed delays in an event-based simulation of service execution in the network. To model the offered load we utilise a Poisson arrival process to generate data transfer requests in the SCTP-user application at the SCP node. By first solving our analytic model of the transport network we can determine the mean arrival rate required to simulate an offered load of 1 Erlang. The IN service application delays measured in the event based simulation include queuing at the ORB/SCTP output buffers rather than just network delays as was the case with earlier simulations. We validate our extensions to the Bellcore model by varying the offered load and comparing the model predictions for link output delay with measurements performed on our simulation. Each reference load was repeated for a number of different network scenarios and all simulations were run for a large number of congestion cycles to enable the system to settle to equilibrium.

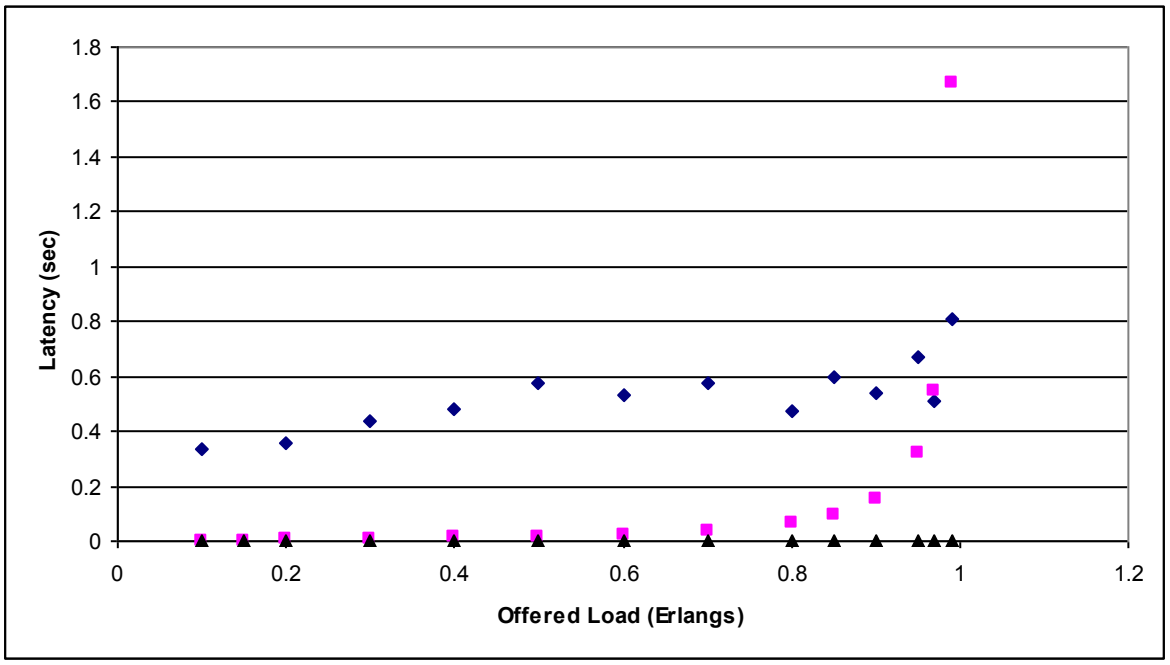


Figure 6.18: Comparison of Model and Simulation Results for DIN SCP Link Delays for a Varying Load

In the figure above we plot the enhanced model estimate (■) from (6.34) and (6.38), the original model estimate for this scenario (▲) and the delays measured in simulation studies (◆). For all loads above 0.2 Erlang the enhanced model provides a better estimate of the actual delays in the network. However the effects of queuing in the network still dominate so the measured transfer delays are still above the enhanced model's estimates. At a load value of 0.99 Erlang the enhanced model starts to predict larger queuing delays than are observed in the network. However at our typical operating threshold of 0.8 Erlang the enhanced model produces a useful correction to the previous model's estimate.

6.4.2. Traffic Engineering Applications in SCTP Networks

The results of our transport network model are immediately applicable to a number of SCTP traffic engineering tasks. In this section we briefly discuss these applications.

The most basic way in which the results generated by our model can be applied is to network planning tasks such as identification of the likely behaviour of given network topology and route scenarios. The model as developed here provides information on both the demands on the network links and routers (aggregate throughputs) and the observed performance (throughput, latency, loss rate, rtt) for applications using the transport network. The availability of estimates for the performance observed by applications is an important difference between this approach and techniques that focus on the network (IP) layer.

The relative speed of computation for a wide range of network sizes means that a number of different scenarios can be rapidly evaluated. For example if performing network dimensioning or provisioning the model can be used to estimate the impact on delay and loss of increasing link capacity. The availability of route-specific measures ensures that different routing policies in a network can be evaluated, as can the impact of establishing new flows in an existing network. One important new result of our work is the explicit calculation of the load on links in the secondary network during normal operation of the primary network. This will help SCTP network planners to design efficient secondary networks. This is because due to SCTP dynamics the presence of an overloaded secondary network will degrade the observed performance of flows that take place primarily on the primary network.

Given the sensitivity of SCTP/TCP-style congestion controls to bottleneck links in a network a major task for SCTP/IP or TCP/IP traffic engineering is the identification of these links. As our model gives an estimate of the behaviour of each link in terms of the packet loss rate that it produces this makes the identification of such links trivial.

One important deficiency of TCP that has been addressed in the SCTP specification is the ability to configure the protocol parameters on a per session basis rather than per network node. Given the

outputs of our transport network model for average throughput on a given path we can establish the desired send rates of SCTP endpoints on a per destination basis. We also know from the work of *Ott et al.* [MSMO97, MBO99] that in equilibrium SCTP/TCP sources have relatively small variances in the distribution of their send rates (again estimates for these are calculable from our model output for observed loss rates). This information would allow the definition of per-destination policies on peak send rates for each node to ensure maximum network stability. In the absence of congestion peak send rates for SCTP sources are limited by their receiver windows. Coupled with the ability of SCTP nodes to set this protocol parameter on a per session basis we have the basis of endpoint rather than router controlled admission control in the network. Thus QoS guarantees would be partially enforced by endpoint rather than network behaviour. This is particularly applicable to relatively static network configurations in tightly controlled environments such as private signalling networks.

Finally our model outputs can be used to establish more traditional admission control policies in DiffServ and IntServ networks. For example our model outputs could be used in a static way to set average packet loss rates or throughputs for individual routes as part of a SLA framework. It is also possible to envisage a more dynamic use of the model whereby new flow admission requests are evaluated (via execution of the model) in terms of their likely impact on the observed performance for all routes in the network. In this scenario the relatively low computational cost of the solutions generated by our model are particularly significant.

6.5. Summary and Conclusions

Chapter 5 introduced and evaluated a model for the performance of multi-homed SCTP flows along a single network route. In this chapter we have specified extensions of the original analytic model, which:

- **Allow networks of competing multi-homed SCTP flows to be modelled.** Previously only the characteristics of an abstract route through the network were estimated by the model, this has been extended to a full characterisation of all competing flows for a specific topology and route set. Thus the interactions of competing traffic in the network are explicitly modelled and the impact on all flows and links is simultaneously evaluated. This enables rapid exploration of the impact of dimensioning decisions or risk assessments to be carried out on proposed network extensions.
- **Enable the estimation of network performance solely from readily available physical characteristics of the network elements and links**. This addresses a disadvantage of the single route model whereby the packet loss rate on a route was an input to the model solution. Although this parameter can be estimated from observations it is not a predefined property of a network infrastructure. Instead it depends on the network, the active flows and their behaviour. By examining the whole network of competing flows it is possible for the model to naturally estimate packet loss rates for all of the active routes. Hence the enhanced model can produce predictions

with just the characteristics that are available to network planners or mangers.

- **Give results that can be used for both network and application deployment planning**. Careful refinement of the model and demonstrations of its applications show how the model can be applied to both estimating the demands on the network infrastructure and the typical performance observed by applications using the transport infrastructure.
- **Provide an estimate of the likely performance effects of the deployment of combined DIN/SCTP networks**.

Together these enhancements transform our original model into a comprehensive set of network planning tools for DIN over SCTP networks. These results are also applicable to a broad range of SCTP-user applications.

The first part of this chapter describes our enhanced model for networks of multi-homed SCTP flows. In common with our approach in Chapter 5 the multi-homed network is partitioned into primary and secondary networks. We first estimate the properties of flows in the primary network by numerically finding the fixed point of the system of non-linear equations governing the dynamic behaviour of SCTP congestion window sizes in that network. This establishes the amount of traffic transmitted on the secondary network, which is also solved through the application of fixed-point methods. The combination of primary and secondary network results allows us to estimate the aggregate performance properties of the combined network as seen by multi-homed SCTP applications such as DIN nodes.

This multi-homed network model was then evaluated against measurements from event-based simulations for a large number of network configurations, topologies and route patterns. These network scenarios included both traditional IN-style configurations and configurations related to the deployment of the DIN. This evaluation has shown that the model's aggregate throughput predictions are accurate to within a factor of two in the vast majority of cases and within order of magnitude for all cases. The aggregate latency predictions of the model have poorer accuracy with typical results being within an order of magnitude but occasional cases, especially those involving large amounts of queuing in the network, are less accurate than this. The model could be extended to include queuing approximations for the network but at a cost in the complexity and flexibility of the method. Earlier work [Rou01] has demonstrated that a number of heuristic corrections to the model's estimates are possible but we instead concentrate on the base accuracy of the method here. In all cases the computational efficiency of the method compared to simulation was demonstrated with execution times of less than 1 second required in all cases.

The application of the extended model to network planning was then discussed and we presented an extension of the standard Bellcore model for SCP performance evaluation to cover the case of the DIN over a SCTP transport network. This treatment was quite general and is applicable to a wide range of

applications or protocols using SCTP. This combined model of DIN SCP performance over a SCTP network was then evaluated for a range of service traffic loads. The results of this evaluation showed an improvement in the predictive power of the model for observed network latency.

Finally there was a brief discussion of ways in which the model could be applied to SCTP traffic engineering tasks. This concentrated in ways which the model's predictions could be used to dimension or configure the network infrastructure itself.

Chapter 7. Conclusions and Topics for Further Research

In this chapter we provide in §7.1 and §7.2 a summary of the conclusions and main contributions of this work. In the remaining sections we outline further areas of potential development. §7.2 addresses possible extensions to the architecture and models developed, §7.3 discusses the application of the work done so far to new application areas and finally §7.4 includes some ideas for the potential impact of this work on broader topics research topics related to next generation telecommunications service networks.

7.1. Conclusions

Telecommunications service providers have a set of service offerings that are daily growing in complexity, heterogeneity and vulnerability to increased end-user expectations based on the ubiquitous presence of advanced data communications services. This has increased the cost of ownership of inflexible service platforms based on traditional telephony models of service delivery. One solution to this problem is the deployment of more flexible platforms based on ICT principles such as middleware that excel as integration technologies. However a vital concern for such platforms is that they have both the flexibility to incorporate new service technologies and that they can transparently inter-work with the existing telecommunications infrastructure. We have presented such an architecture in the form of the DIN. This work has been accepted as an international standard by the OMG.

However implementation experience with the DIN has taught us that such a distributed platform is vulnerable to failings at the transport layer. Hence a thorough investigation of the properties of the most likely candidate for deployment in the telecommunications network, SCTP, has allowed us to identify and fix a number of failings in that protocol. These enhancements to the protocol have been accepted by the IETF, will be included in the next release of the SCTP standard and are already included in the reference implementation of SCTP.

We have developed the first analytic model of multi-homed SCTP associations on a single route. This model has been shown to produce useful results for both network performance as perceived by applications and network load over a wide range of operating conditions. This model was then extended to provide the first analytic model of networks of competing multi-homed SCTP flows. Finally the transport model was combined with a standard model of IN nodes to predict likely DIN performance

over a SCTP network. This work has illustrated how combined DIN/SCTP signalling and advanced services networks are a viable architecture for next-generation service networks. The applicability of our work on SCTP transport networks is much broader than just the DIN and it will also contribute to initiatives such as the International Packet Switching Consortium for IP-enabled telephony and the general development of the Internet infrastructure.

The new analytic models developed during our work on the performance evaluation of SCTP have allowed us to develop an efficient approach for the dimensioning and planning of networks of competing multi-homed SCTP flows. These techniques provide rapid, scaleable solution. Tools like this are a requirement for the widespread adoption of new protocols and architectures.

7.2. Main Contributions

The work presented here has addressed the topic of flexible next-generation telecommunications service delivery platforms based on the CORBA middleware platform and SCTP signalling transport. The implicit increase in distribution of service logic in this system and the consequent reliance on inter-node communications has required us to take a broader view of what constitutes the service platform itself. Hence the central role of signalling transport in the form of the heretofore relatively untested SCTP protocol.

The main contributions are:

- **A new middleware-based architecture for Distributed IN (DIN)**. This architecture enables the evolution of current telecommunications service networks towards flexible distributed service platforms which have stronger integration with new service delivery mechanisms and network management systems. It addresses key weaknesses in the current IN in terms of cost of ownership, flexibility of implementation, isolation from the TMN and dependency on specialised equipment vendors. It enables a wide range of new deployment scenarios and supports current industry trends towards ICT integration and third party service provision. This architecture has been standardised by the OMG Telecommunications Domain Task Force. It incorporates:
  - A survey of the public literature relating to the design and deployment of CORBA-based telecommunications service platforms, together with an indication of some issues not addressed therein;
  - An architectural framework for the implementation of both native CORBA TC-User Application Entities (SSPs, SCPs) and CORBA to TC-User Gateways;
  - A specification translation algorithm for the conversion of ITU ASN.1 TC-User application protocols into OMG IDL object interface specifications;
  - An interaction translation specification that deals with how to name, locate and interact with TC-User implementations in the CORBA domain in a way that provides location transparency

for both native CORBA implementations and traditional TC-User implementations in the SS.7 domain;

- A set of TC PDU-oriented Interfaces that define an API that standardises access to SS.7 protocol stacks in the CORBA domain, either for use at CORBA to TC-User Gateways or by CORBA applications that wish to use TC/SS.7 message transport;
- A survey of deployment scenarios and implementation experiences;

- **A comprehensive examination and appraisal of the performance of the SCTP signalling transport protocol**. This work has identified a number of flaws with the current protocol's congestion control specification leading to a number of suggested improvements in the operation of the protocol to fix defects and improve performance. These recommendations have been accepted by the IETF and are included in the working draft for the next official version of the specification. In addition we developed the first analytic model for the throughput of a single multi-homed SCTP association. This model has been evaluated over a wide range of network conditions and shown to produce a reliable estimate of performance on a single association. This work incorporates:
  - A survey of the literature relating to aspects of the operation and performance of congestion control as implemented in SCTP and its precursor TCP;
  - An analysis of the operation of single-homed SCTP congestion controls using experimental results and both analytic and simulation models under a variety of network conditions;
  - The development and testing of an analytic model for the throughput of a single multi-homed SCTP association;
- **Development of a framework for dimensioning of competing multi-homed SCTP flows**. This is the first analytic framework for evaluation of the performance of networks of arbitrary topologies of competing multi-homed SCTP associations. It has been shown to be computationally efficient and capable of rapidly producing results with an acceptable degree of accuracy for network dimensioning tasks. We have also illustrated how it can be combined with a modified version of standard SCP models to produce estimates of DIN application performance. This work incorporates:
  - Enhancements to our analytic model for the throughput of a single multi-homed association to enable the modelling of networks of competing flows;
  - A set of signalling network scenarios for DIN over SCTP networks;
  - An analysis of the performance of networks of DIN nodes utilising multi-homed SCTP associations for signalling transport;

7.3. Future Extensions of DIN Architecture and Network Model

The DIN architecture and network transport model specified in chapters 4 to 6 delivers a basis for the

development of the traditional IN and a means of performing network analysis and planning in these new environments. However there are many opportunities for new topics of research that leverage the work already completed. Three such topics are discussed here: enhancements to the DIN architecture to make it more comprehensive, the combination of our network transport model with independent work on optimal object placement in a CORBA-based environment and finally extensions to our network model.

7.3.1. DIN Enhancements

The DIN as standardised enables the construction of CORBA-based IN service entities and gateways that mediate between such entities and traditional IN entities. However the constantly evolving telecommunications service environment has many other important actors that could be embraced by a broader CORBA-based telecommunications service architecture. Essentially this involves the broadening of the scope of the DIN to include both existing and emerging service delivery mechanisms and integration with network management functions and third party service access frameworks.

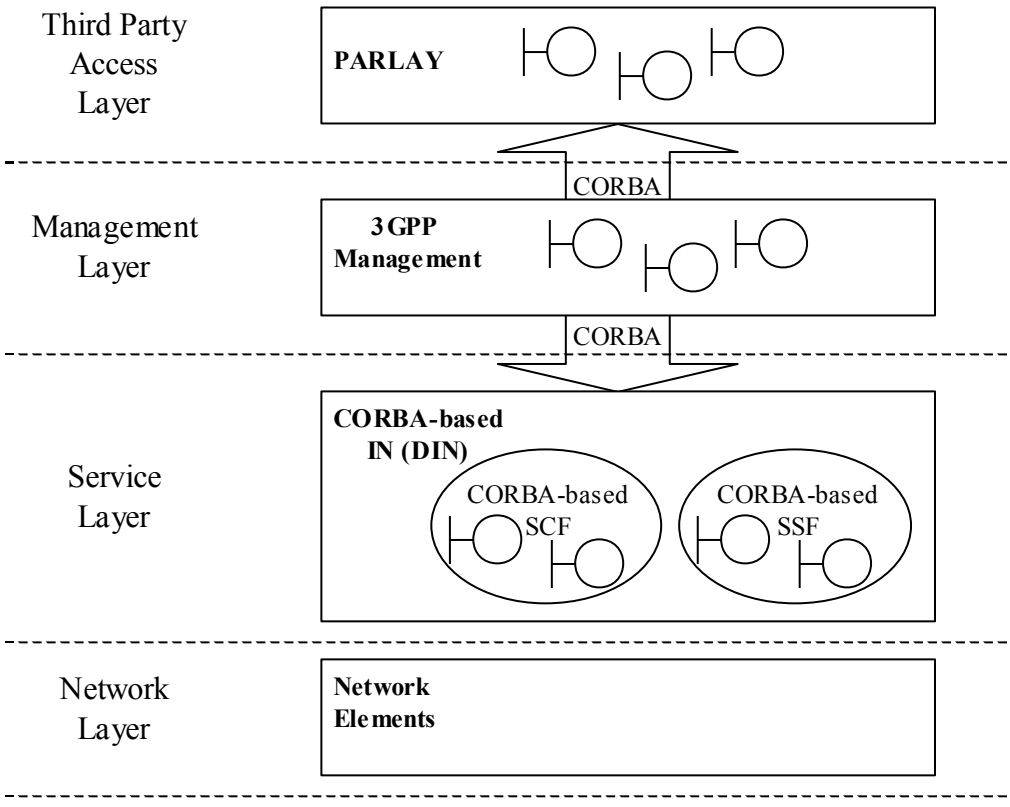


Figure 4.1: General CORBA-based Telecommunications Service Framework

If we consider again our view of the telecommunications service framework presented in figure 4.1 (repeated above for convenience) we see four layers linked by a common CORBA-based communications infrastructure. This flexible approach is required to maximise the opportunity to satisfy

customer requirements for new services and to enable the integration of new technologies as they mature. It is important to recall that although only a CORBA-based view of service and network elements is illustrated, that in fact access to traditional implementations is also catered for through the implicit specification of gateways as was done in the DIN case.

7.3.1.1. Third Party Access Layer

The requirements for this layer have been identified by previous work on the TINA Broker service [TINA] and continued in both the Parlay Consortium [Pa03] and the OMG in their work on the Telecommunications Service Access and Subscription service [TSAS]. Work in this area is already at an advanced stage and given the developments envisaged at the lower layers in our architecture it is really only implementation issues that still need to be solved. The existence of the DIN specification in tandem with the existing work in this area in fact enables the construction of open third party service access gateways (currently limited to IN services) today.

7.3.1.2. Management Layer

Given the early evolution of management systems to an OO approach to data modelling CORBA has long been seen as the natural medium for exchange of management information. This has been reflected in CORBA-based telecommunications management standardisation initiatives in the TMF [TMF], ITU [ITUCOR], OMG [OMGMGT] and more recently the 3GPP [MGT]. In addition there has been voluminous research on telecommunications service-specific management entities such as CORBA-based service creation environments [LKH99].

An aspect of next generation service networks that management systems will have to adapt to the new realities is in the definition of what is a service. In the future one can see that service deployment, subscription, composition and configuration are going to be more dynamic. Many services will be composed by end-users through the third party layer and will consist of the collaboration of many elemental services based on both data and voice resources. Thus many traditional network support functions, for example the provision of data storage, may themselves become elemental services that are incorporated into end-user composite services. This requires the management system to change its view of what a service is and more particularly the granularity at which services or elemental services are defined. The management system would naturally be the source of service or elemental service availability information to the third party access layer hence definition of interfaces or procedures to enable this are an important work item. In essence the two network control paradigms of (real-time) service control and (non-realtime) network management need to merge into one continuous spectrum of network control functions.

7.3.1.3. Service Layer

The DIN architecture forms a strong basis for deploying a CORBA-based service layer. This addresses traditional IN services both implemented on a CORBA platform and via gateways in traditional telecommunications service platforms. Other traditional telephony service protocols such as GSM-MAP are also catered for by the DIN architecture (see below). This is insufficient for the needs of modern and next-generation service networks, which must incorporate traditional IN and combined PSTN/Internet telephony (or multimedia) services. In fact the ideal open service platform would also allow for the integration of as yet undefined service technologies as they emerge.

As illustrated in figure 4.1 a necessary requirement for the integration of new IP-based service technologies would be the development of a CORBA-based framework for SIP and related services. This would take the form of a set of specification and interaction translation rules like those already developed for the IN. The scope of this work would include SIP, SDP, PINT and SPIRITS. All of these protocols are undergoing continued development within the IETF. The goal of this work would be to enable the creation of CORBA-based SIP entities that could transparently collaborate with either traditional SIP entities via gateways or each other. Ideally composite services with both SIP and traditional IN aspects could then execute on a common CORBA-based platform.

Work within the 3GPP has suggested that a SIP/SCTP network be built for service delivery. If this were realised then our CORBA/SCTP network would naturally fit into the operator model of a common switching fabric for control and data channels, even in networks with both CORBA-based and traditional SIP elements.

Finally ETSI TIPHON style services could be integrated into such a service layer via similar work to SIP/DIN performed on H.323 style IP telephony. In the medium term it is likely that a SIP-based approach would be a more useful addition to the architecture.

7.3.1.4. Network Layer

The inclusion of the network layer in our model may seem superfluous in a service-oriented architecture – wouldn't IN protocols give us sufficient means to control the switching infrastructure? However in practice many current IN control nodes (SCPs) include some sort of network signalling capability such as Q.931 (or variants) support. This is especially useful in the control of intelligent peripheral nodes. Hence the development of CORBA-based interfaces for Q.931 (and variants) interactions between the service layer and network layer is a desirable work item. We can also see the analogue of this capability in the work of the Xbind/OPENSIG research project, which developed such control interfaces for ATM switching nodes and the non-CORBA based work of the Softswitch Consortium (now International Packet Communications Consortium or IPCC). The IETF's MGCP is another important protocol requiring the definition of CORBA interfaces in this layer to maximise the utility of our expanded architecture.

Hence the definition of various signalling control interfaces for elements of the switching network would be of great utility in realising an open, CORBA-based service architecture. In addition this work would ease the opening up of the network to next generation composite services that may need to access elemental services at a finer granularity than has been traditionally the case.

7.3.2. Combining DIN Optimal Object Placement with Network Model

Independent work on load control for DIN systems has developed a method for optimising the placement of software objects representing DIN service components on network nodes [MCA]. This optimisation addresses the performance effects of distributed communication between collaborating objects in the CORBA domain whether resident at an IN-CORBA gateway or as native CORBA objects implementing DIN interfaces and acting as service components. The performance impacts of different service types and mixes of service requests are also considered when optimising object placement within the network's processing nodes.

A Mixed Integer Programming (MIP) flow problem formulation is used to yield the distribution of DIN service objects for which the maximum service request arrival intensities can be handled by the network of finite capacity computational nodes constrained by object installation costs. Given that actual service arrival intensities will vary over time from the optimum distribution design point derived above, a load control and distribution algorithm for the resultant network is then developed to optimise routing between object instances whilst maximising network revenue. This is formulated as a Linear Programming problem.

There are several ways in which the above work could be synthesized with our work on SCTP transport network modelling in the DIN. Two such areas for further work are described in the following sub-sections.

7.3.2.1. Generating Scenarios

In chapter 6 we constrained our investigation of DIN network flows over a SCTP transport to only consider basic cases where DIN entities mapped directly onto traditional IN functional entities such as the SCF and SSF or minor amounts of distribution for service logic. As part of the development of algorithms for optimal placement of DIN computational objects McArdle has decomposed these functional entities into their constituent fine-grained computational objects. All of these computational objects, some of which form service components, are potentially distributed over a number of network nodes. This distribution will in turn shape the traffic transported by the SCTP network.

Hence we may use the solutions to McArdle's optimal object distributions as new scenarios to be evaluated with our network transport model. In general these solutions will involve a much larger number of network nodes as the functionality previously encapsulated within a single node, for example

a SCP, will instead be distributed over a number of nodes. Thus more complex patterns of traffic routes can be expected in the network and these are worthy of investigation. In summary, McArdle's work will provide new network scenarios to be evaluated using our model of networks of SCTP flows. This will help us gain an understanding of the DIN behaviour and better evaluate our network model.

7.3.2.2. Enhancing Network Planning/Dimensioning

McArdle's work aids network planning by defining optimal object placement within the network. This is done in terms of minimising communications costs and maximising the service arrival intensity that can be handled by a given set of networked computational nodes. However the model makes certain assumptions about the network characteristics that don't take into account the likely performance impacts of the presence of competing adaptive flows. In contrast our SCTP transport network model explicitly includes the aggregate effects of all flows on the network to determine the likely network performance observed on a particular route through the network.

Thus it seems likely that some combination of the two models would produce a more accurate picture of the performance consequences of various object distributions on the network. An initial line of investigation could be to try and include our network transport model as a constraint on the results of the optimal object placement algorithm. Hence as part of the optimisation process bottleneck routers would be identified and instead the optimisation would favour alternative object distributions that controlled network load as well as maximised likely service completion rates for the network based on load at the nodes themselves.

7.3.3. Extending SCTP Transport Analytic Model

In Chapters 5 and 6 we developed our analytic model of networks of flows from multi-homed SCTP endpoints. This has been tested under a wide variety of network topologies and our results are sufficiently accurate for typical network dimensioning tasks. However there are a number of areas in which the model could be extended. Some of these are discussed in the following sub-sections.

7.3.3.1. Modelling an Association with More Than Two Paths

The SCTP protocol supports multi-homed endpoints with an arbitrary number of paths. The work presented here has concentrated on the single and dual-homed cases. In principle it would be possible to extend the network throughput model to accommodate an arbitrary number of paths. One important aspect of the design of any such model would be an investigation of the effects of differing retransmission path selection policies or strategies. In contrast to the case of single or dual-homed SCTP endpoints, retransmission path selection policy for many-homed endpoints is an area that the current SCTP specification leaves up to the implementation. Therefore the development and evaluation

of a range of retransmission path selection policies or strategies would be of great use to the research community. Treating many-homed SCTP endpoints could also pose significant new questions about SCTP's congestion control algorithms and how best to optimise the performance of SCTP in these environments.

7.3.3.2. Modelling Arbitrary Combinations of Primary and Secondary Routes

Our current SCTP network transport model treats the primary and secondary networks separately in order to simplify the problem formulation and reduce the computational complexity of the numerical solution. However work by Fireau *et al.* [Fir01] has illustrated a possible method by which networks of independent persistent and non-persistent TCP sources may be combined in a single treatment. Of course the dual-homed SCTP case is not so simple due to the dependencies between the primary and secondary sources. Nonetheless it is possible in principle to envisage a solvable network problem consisting of both primary and secondary sources sharing a common network infrastructure. Perhaps an iterative form of solution would be required with initial estimates being used for secondary traffic that would then converge towards a fixed point based on the solutions of our expressions for primary route traffic. This would allow for more realistic signalling network topologies including cross-links to be investigated.

7.3.3.3. Modelling Network Queuing Delays

In high bandwidth, low delay networks a principal contributor to RTT is queuing delays in busy routers. The current analysis of SCTP networks is based on the simplifying assumption that transmission delays dominate in the scenarios studied. This approach follows that of Roughnan et al [Rou01] who showed that while simple queuing theory assumptions can be useful as zeroth order model for queuing in SCTP/TCP over IP networks that ultimately they are not very accurate and do not lead to increased robustness of solutions. This is due to the well-documented difficulties with representing the positive feedback effects of TCP-style congestion controls in queuing systems [Arv01]. However this is certainly an area for further study.

7.4. New Applications for DIN Architecture and Network Model

In the previous section we examined some of the natural extensions that could be made to our existing model and architecture. In this section we discuss some new applications of the work already completed.

7.4.1. DIN Architecture

The DIN architecture as standardised provides a basis for CORBA-based service platforms that support any TC-User protocol such as ITU or ETSI IN capability sets, X.500 Directory Access (used in the

USA for SDP access) or GSM-MAP. Hence its applicability is broader than just the traditional IN.

In addition the model of distributed IN services embedded in the current IDL specifications could be converted into a UML specification that would broaden the appeal and applicability of the model. This would enable the development of both automated IN application development, using the OMG's Model Driven Architecture, and the implementation of platforms based on new middleware technologies such as Web Services. This would allow the reuse of the meta-model of IN services already developed.

7.4.2. SCTP Transport Network Model

The current work has defined a comprehensive multi-homed SCTP transport network model however it has only been applied to DIN-style networks. The general applicability of SCTP as a next-generation Internet transport protocol replacing TCP has been recognised by the IETF community and many application research areas are now investigating the use of SCTP. This leads to a large number of potential applications for our SCTP transport network model that reaches far beyond the limited remit of telecommunications signalling. We explore a few of these topics in the following sub-sections.

7.4.2.1. Modelling SCTP/SIP Networks

IP telephony architectures based on SIP and related protocols such as PINT and SPIRITS are proliferating. The SIP community has already envisaged the potential gains of SCTP as opposed to TCP signalling transport. However there has been little investigation of how the transport requirements of SIP could be used to refine the services that SCTP offers. Thus the focus of work in this area could be to investigate what the application protocol requires rather than what the transport protocol already provides. It is certainly the case that the origins of SCTP requirements in signalling applications should already provide useful features for SIP applications.

The decentralised SIP architecture implies very different traffic demands and typical network topologies than the traditionally more centralised IN or DIN scenarios examined so far. Thus the investigation of SIP networks would provide a source of new scenarios for evaluation of both our analytic model and the SCTP protocol.

One of the remaining obstacles to wide scale commercial deployment of SIP or other V/MoIP solutions is performance concerns. Hence the development of combined SIP over SCTP models could help to address this issue. In addition implementers will need to evaluate the costs and benefits of deploying platforms that make use of SIP over SCTP rather than over TCP. Our existing model could form part of the basis of that evaluation.

7.4.2.2. Evaluation of Signalling Gateways

The International Packet Communications Consortium has developed an evolving architecture for VoIP

networks that is largely based on IETF protocol specifications rather than the ITU H.323 focused work of ETSI TIPHON. These represent two coherent frameworks for the deployment of VoIP services. A key element in both architectures is the signalling gateway functional entity. This acts as a bridge between traditional SS.7 signalling networks and IP-based service/signalling networks. SCTP is the most likely candidate for transport of signalling information in the IP domain.

Hence the availability of our SCTP network models could contribute towards both the continued development of these architectures and the actual practice of designing or deploying these networks. Architectural development would be aided by combing our SCTP model with typical network topologies and suitable models of higher layer protocols or computational tasks at the gateway to investigate the performance of current or new proposals in this area. In comparison with the relatively well-known properties of the SS.7 network there remains a large amount of work to be done on IP-based signalling solutions. The presence of a SCTP network in these environments would make our model a natural choice when performing network dimensioning or planning for actual deployments.

This work would also provide yet more scenarios for the evaluation and development of our SCTP models. As the idea of the combined service and signalling network is gaining in importance for IP environments it would also be interesting if our SCTP network model could be extended to also include TCP flows (fairly trivial) and also non-adaptive (UDP) flows such as media traffic. Fireau *et al.* have employed a slightly different approach to TCP network modelling [Fir01] which includes non-adaptive flows and perhaps that work could be combined with our own to allow solutions for systems of competing SCTP. TCP and UDP flows.

7.4.2.3. Modelling Traditional IN over SCTP Networks

The IETF SIGTRAN group that developed the original SCTP specification has continued to produce a family of specifications for IP-based signalling transport. This essentially consists of a set of SS.7 layer adaptation layer protocols that reside above SCTP that provides transport services for them. Thus there is a MTP1 adaptation layer protocol which provides the MTP1 services over a SCTP/IP transport network and similar protocols for each of the SS.7 layers up to SCCP. When finished this family of protocols will provide a basis for deploying standard telecommunications signalling applications on a SCTP/IP network infrastructure and building gateways that inter-work between the SS.7 and SCTP/IP environments.

Both of these applications raise important performance issues that must be addressed before such a system can realistically be deployed. This is especially significant in the case of SS7-SIGTRAN gateways, as such systems will necessarily be connected to SS.7 networks. SS.7 includes many performance (timing) requirements as part of its specification. This means that integration with best-effort networks, such as IP, may introduce recurrent error conditions due to the expiry of SS.7 timers.

Even in a "pure" IP environment with no SS.7 connectivity there may be similar failures due the requirements imposed by applications that have been developed for the SS.7 environment.

In essence there is still uncertainty that adaptation layers operating above a SCTP/IP network can actually deliver an acceptable facsimile of SS.7 layer services, or at least that such adaptation layers will have to be very carefully designed to do so. The models developed as part of the current work could be used to investigate these issues, especially when combined with new models of the SIGTRAN adaptation layer protocols. This work would help to validate the SIGTRAN architecture and would undoubtedly lead to improvements or new work items for the standardisation process.

7.5. Broader Research Topics for Next Generation Service Networks

All of the work presented here has been carried out with the aim of developing next-generation telecommunications service platforms. The application of modern distributed systems technology to the service platform it has made it necessary to consider the signalling transport network as a part of the "platform" in a broad sense. This is due to two considerations, firstly the traditional specification of real-time service control and execution deadlines for the combined system of platform and network and secondly the increased distribution of service control logic among a number of collaborating nodes connected by the signalling transport network. These developments are primarily driven by two concerns, increasing the flexibility and ease of integration of the service platform into the wider data/telecommunications network. An underlying feature of both of these concerns is the integration of traditional data communications technology into the heterogeneous telecommunications network. Doubtless future directions for research into next generation service networks will continue to be driven by these underlying concerns.

One relevant emerging technology area is that of peer-to-peer (P2P) or Grid computing. Traditionally this has been concerned with the parallel application of wide-area computing resources to complex problems in the field of scientific computing however the application of this technology to general computing tasks, such as service delivery, is now mooted. The grid platforms themselves are still in an embryonic stage of development and many of their promises of ubiquitous access to computing resources sound familiar but increasingly major equipment vendors are taking note. It may be the case that we see in Grid technologies the new platform for telecommunications services in the medium term. Grid evangelists would argue that we also see the new paradigm for service development but this is doubtless a longer-term vision. It is more likely, at least in the medium term, that the Grid model for service development, deployment and delivery would have to encapsulate traditional service models in order to interoperate with existing service networks. The DIN architecture is ideally placed to influence this work, at least for Grid frameworks based on CORBA.

If we did have a global grid of computational, data, application, information and knowledge services available then exactly how would telecommunications services fit into this framework? How could telecommunications service providers leverage the framework to deliver services to customers? Could CORBA continue to be utilised as a user-level middleware (as it is in some prototype Grid frameworks) or will new middleware be required? Given the global scope of Grid computing SCTP would doubtless be included as a transport network between some grid nodes. How would this network resource be managed by Grid management entities? Resource management and scheduling has been identified, along with satisfying user-specified QoS as key to the success of Grid technologies. Perhaps the SCTP models developed in the current work could be usefully applied in that context.

Publications and Reports Related to this Thesis

This thesis contains material from a number of publications and reports. These are listed in chronological order as follows:

1. N. Mitra, R. Brennan, "Design of the CORBA/TC Interworking Gateway". In Han Zuidweg et al. (eds.), In Proceedings of the 6th International Conference on Intelligence in Services and Networks, (IS&N'99), Barcelona, pp 84-100, April 1999.
2. C. McArdle, R. Brennan, N. Jones, J. Vasic and T. Curran, "Implementation Experience with the OMG IN/CORBA Interworking Specification". In Proceedings of the 6th International Conference on Intelligence in Networks (ICIN'00), Bordeaux, March 2000.
3. R. Brennan, B. Jennings, C. McArdle and T. Curran, "Evolutionary Trends in Intelligent Networks", IEEE Communications Magazine, Volume 38, No. 6, pp86-93, June 2000.
4. R. Brennan (Ed.), "CORBA/TC Interworking and SCCP Inter-ORB Protocol Specification", OMG document formal/01-01-01, pub. Object Management Group, January 2001. Available at http://www.omg.org
5. R. Brennan, B. Jennings, C. McArdle and T. Curran, "Intelligent Networks: A Discussion of Current Developments", In Gerald Grant (ed.): Managing Telecommunications and Networking Technologies in the 21st Century: Issues and Trends, pub. Idea Group Publishing, pp1-20, April 2001.
6. R. Brennan and T. Curran, "SCTP Congestion Control: Initial Simulation Studies", in J. Moreira de Souza, N. L. S da Fonseca and E. A. de Souza e Silva (Eds.) Proceedings of 17th International Teletrafic Congress (ITC-17), pub. Elsevier Science B. V., vol. 4, pp 843-856, Salvador da Bahia, Brazil, September 2001.
7. T. Ravier, R. Brennan and T. Curran, "Experimental Studies of SCTP multi-homing", First Joint IEI/IEE Symposium on Telecommunications Systems Research, Dublin, December 2001.
8. R. Brennan and T. Ravier, "TCP Analytic Models Applied to SCTP: An Experimental Evaluation". in the proceedings of Information Technology and Telecommunications 2002 (IT&T '02), pp92-100, Waterford, October 2002.
9. R. Stewart, L. Ong, et al., "SCTP Implementor's Guide", IETF document number draft-tswg-sctpimpguide-10, work in progress, available at http://www.ietf.org, September 2003.

At the time of writing the following paper has been submitted for publication:

10. R. Brennan and T. Curran, "Network Performance for Multi-Homed SCTP Networks", submitted for publication in the Special Issue of Elsevier Computer Communications on IP Networks, Summer 2005.

References

[3GSCTP] 3GPP, “TR 29.903 V5.0.0 Feasibility Study on SS7 signalling transport in the core network with SCCP-User Adaptation Layer (SUA)”, 2001, available at http://www.3gpp.org

[AAI02] R. Alamgir, M. Atiquzzaman, and W. Ivancic, "Effect of Congestion Control on the Performance of TCP and SCTP over Satellite Networks", NASA Earth Science Technology Conference, Pasadena, CA, June 2002. Available at http://www.cs.ou.edu/~atiq/publications.html.

[AI03] M. Atiquzzaman and W. Ivancic, "Evaluation of SCTP Multistreaming over Satellite Links", 12th International Conference on Computer Communications and Networks, Dallas, TX, Oct 20-22, 2003. Available at http://www.cs.ou.edu/~atiq/publications.html.

[AL97] Z. Canela, H. Zuidweg (Eds), “IN using CORBA – Response to the OMG Request for Information”, Ref ULT/C/970121, 1997, Available at http://www.omg.org

[All98] M. Allman. “On the Generation and Use of TCP Acknowledgments”. ACM Computer Communication Review, vol. 28, no. 5, pp 4-21, October 1998

[All99] M. Allman, “TCP Byte Counting Refinements”, ACM SIGCOMM Computer Communication Review, vol. 29, no. 3, pp14-22, July 1999.

[All99b] M. Allman and A. Falk, “On the Effective Evaluation of TCP”. ACM SIGCOMM Computer Communication Review, vol. 29, no. 5, pp59-70, October 1999.

[Amoeba] A. S. Tanenbaum and S. Mullender, "An overview of the Amoeba distributed operating system," Operating Systems Review, vol. 15, no. 3, pp. 51-64, July 1981.

[APS99] M. Allman, V. Paxson, W. Stevens, “RFC 2581: TCP Congestion Control”, The Internet Society, 1999.

[Arv01] A. Arvidsson, M. Roughan and T. Ryden, “On the Origins of Long Range Dependence in TCP Traffic”, ”, in in J. Moreira de Souza, N. L. S da Fonseca and E. A. de Souza e Silva (Eds.) Proceedings of the International Teletraffic Congress (ITC-17), pp895-906, Salvador, Brazil, September 2001.

[Arv99] A. Arvidsson and P. Karlsson, “On traffic models for TCP/IP”, in Proceedings of the International Teletraffic Congress – ITC16, Edinburgh, UK, June 1999.

[Bell97] Bellcore, "Advanced Intelligent Network Generic Requirements (AINGR): Switch – Service Control Point(SCP)/ Adjunct Interface, ", GR-1299-CORE, Issue 4, Sept. 1997.

[Booch91] Booch, Grady, “Object Oriented Design with applications”, The Bejamin/Cummings Publishing Company Inc., 1991.

[BR02] R. Brennan and T. Ravier, "TCP Analytic Models Applied to SCTP: An Experimental Evaluation". in the proceedings of Information Technology and Telecommunications 2002 (IT&T '02), pp92-100, Waterford, October 2002..

[BT01, Bu01] T. Bu and D. Towsley, "Fixed point approximations for TCP behaviour in an AQM network", in ACM SIGMETRICS, Cambridge, USA, June 2001.

[Car00] N. Cardwell, S. Savage, T. Anderson, "Modelling TCP Latency", IEEE INFOCOM 2000, Tel Aviv, Israel, March 2000.

[Car98] M. Carlson, W. Weiss, S. Blake, Z. Wang, D. Black and E. Davies, " RFC 2475: An Architecture for Differentiated Services", The Internet Society, December 1998.

[CHCAF01] P. Conrad, G. Heinz, A. Caro, P. Amer and J. Fiore , "SCTP in Battlefield Networks", MILCOM 2001, Washington, DC, November, 2001

[CIAHS02] A. Caro, J. Iyengar, P. Amer, G. Heinz and R. Stewart, "A Two-level Threshold Recovery Mechanism for SCTP", SCI 2002, Orlando, FL, July, 2002

[CIAHS02a] A. Caro, J. Iyengar, P. Amer, G. Heinz and R. Stewart, "Modeling SCTP Latency with Multihoming and Failovers", unpublished manuscript. Available at: http://www.eecis.udel.edu/~amer/PEL/poc/index.html

[CM02] Cisco and Motorola, "SCTP Reference Implementation", available at: http://www.sctp.refcode.org , 2002

[Coe01] L. Coene et al., "Telephony Signalling Transport over SCTP applicability statement", Internet Draft draft-ietf-sigtran-signalling-over-sctp-applic-09.txt, work in progress, 2003.

[Coe02] L. Coene, "RFC3257: Stream Control Transmission Protocol Applicability Statement", The Internet Society, 2002.

[CORBA] Object Management Group, "Common Object Request Broker Architecture", Version 3.0, OMG , July 2002

[Eri00] Ericsson, "Interim Report on Dimensioning of UMTS Core Networks", 2000

[ETSI] ETSI, "ETS 300 374-1, Intelligent Network (IN); Intelligent Network Capability Set 1 (CS1); Core Intelligent Network Application Protocol (INAP); Part 1: Protocol specification ", ETSI, 1994.

[FA03] S. Fu and M. Atiquzzaman, "Improving End-to-End Throughput of Mobile IP using SCTP", 2003 Workshop on High Performance Switching and Routing, Torino, Italy - June 24-28, 2003, pp. 171-176

[FAI02] S. Fu, M. Atiquzzaman and W. Ivancic, "Effect of Delay Spike on SCTP, TCP Reno, and Eifel in a Wireless Mobile Environment", International Conference on Computer Communications and Networks, Miami, FL, Oct 14-16, 2002, pp. 575-578.

[FF96] K. Fall. and S. Floyd, "Simulation-based Comparisons of Tahoe, Reno and SACK TCP", Computer Communication Review, vol. 25, no. 5, pp5-21, July 1996.

[FHPW00] S. Floyd, M. Handley, J. Padhye and J. Handley, "Equation-based congestion control for unicast applications", in Proceedings of the conference on Applications, Technologies,

Architectures, and Protocols for Computer Communication, pp43-56, Stockholm, Sweden, August 2000.

[Fir00a] V. Firiou and M. Borden, "A study of active queue management for congestion control", in IEEE INFOCOM 2000, pp. 1435-1444, 2000.

[Fir01] V. Firiou, I. Yeom and X. Zhang, "A framework for practical performance evaluation and traffic engineering in IP networks" in IEEE International Conference on Telecommunications, Session WE20, June 2001.

[Fir02] V. Firoiu, J. Le Boudec, D. Towsley, Z. Zhang. , "Theories and Models for Internet Quality of Service". In Proceedings of IEEE, special issue on Internet Technology, vol. 90, no. 9, pp. 1565–1591, Sept. 2002.

[Flo02] S. Floyd and E. Kohler, "Internet Research Needs Better Models", First Workshop on Hot Topics in Networks, Princeton, New Jersey, October 28–29, 2002. Available at: http://www.icir.org/floyd/papers.html

[Flo91] S. Floyd, "Connections with Multiple Congested Gateways in Packet-Switched Networks Part 1: One-way Traffic", Computer Communication Review, Vol.21, No.5, October 1991, p. 30-47

[Flo95] S. Floyd, "TCP and Successive Fast Retransmits", Technical Report, Feb 1995. Available at: ftp://ftp.ee.lbl.gov/papers/fastretrans.ps

[GHJV94] E. Gamma, R. Helm, R. Johnson, J. Vlissides, "Design Patterns – Elements of Reusable Object-Oriented Software", Addison-Wesley, 1994

[Gib00] R. Gibbens, S. Sargood, C. V. Eijl, F. Kelly, H. Azmoodeh, R. Macfadyen, "Fixed-point models for the end-to-end performance analysis of IP networks", in 13th ITC Special Seminar: Internet Traffic Measurement and Modeling, 2000.

[HPF99] M. Handley, J. Padhye, S. Floyd, "TCP Congestion Control Window Validation", Sep. 1999.

[ICAHR02] J. Iyengar, A. Caro, P. Amer, G. Heinz and R. Stewart,"SCTP Congestion Window Overgrowth During Changeover", SCI 2002, Orlando, FL, July, 2002.

[ICAHR03] J. Iyengar, A. Caro, P. Amer, G. Heinz and R. Stewart, "Making SCTP More Robust to Changeover", SPECTS 2003, Montreal, July, 2003

[IPCC] International Packet Communications Consortium, http://www.packetcomm.org/

[IS136] TIA/EIA, "TDMA Cellular/PCS – Radio Interface – Mobile Station – Base Station Compatibility – Digital Control Channel", TIA/EIA/IIS-136.1-A, 1996

[ITU97] ITU-T Rec. Q1221, "Introduction to Intelligent Network Capability Set 2," Sept. 1997

[ITUCOR] ITU, Recommendation M3120, "CORBA generic network and network element level information model", October 2001.

[ITUTele] ITU, "Teletraffic Engineering Handbook", work in progress. Available at http://www.tele.dtu.dk/teletraffic/

[Jac88] Van Jacobson, “Congestion avoidance and control”, in Proceedings ACM SIGCOMM 1988, pp314-329, 1988.

[Jac90] Van Jacobson, "Modified TCP Congestion Avoidance Algorithm", end2end-interest mailing list, April 30, 1990. Available at: ftp://ftp.isi.edu/end2end/end2end-interest-1990.mail

[JAIN] Sun Microsystems, “Java Advanced Intelligent Network”, available at http://www.sunworld.com/swol-10-1998/swol-10-phone.html

[JB88] V. Jacobson and R. Braden “RFC 1072: TCP Extensions for Long-Delay Paths”, The Internet Society, October 1988.

[Jen01] B. Jennings, “Network-oriented Load Control for SS.7/IN”, PhD Thesis, Dublin City University, 2001.

[JIDM] The Open Group, “Preliminary Specification Inter-Domain Management: Specification Translation”, X/Open Document:P509, ISBN:1-85912-150-0

[JST00] A. Jungmaier, M. Schopp and M. Tuxen, “Performance Evaluation of the Stream Control Transmission Protocol”, Proceedings IEEE Conference on High Performance Switching and Routing, pp141-148, Heidelberg, Germany, June 2000.

[Kel86] F. Kelly, “Blocking probabilities in large circuit switched networks”, Advances in Applied Probability, vol. 18, pp. 473-505, 1986.

[Key98] P. Key, “Fixed point methods and congestion pricing for TCP and related schemes”, in Workshop on the modelling of TCP, (Ecole Normale Superieure), 1998.

[LACI03] S. Ladha, P. Amer, A. Caro and J. Iyengar, “Improving file transfer in FCS networks MILCOM 2003, Boston, 10/03.

[Laz95] A.A. Lazar, S. Bhonsle and K.S. Lim, "A Binding Architecture for Multimedia Networks", Journal of Parallel and Distributed Systems, Vol. 30, Number 2, November 1995, pp. 204-216.

[Laz96] A.A. Lazar, K.-S. Lim, "Programmability and Service Creation for Multimedia Networks", Workshop on Multimedia and Collaborative Environments, Fifth IEEE International Symposium on High Performance Distributed Computing (HPDC-5), Syracuse, NY, August 6-9, 1996, pp. 217-223.

[LKH99] F. Lodge, K. Kimbler, M. Hubert, “Alighnment of the TOSCA and SCREEN Approaches to Service Creation”, in Proc. IS&N99, LNCS 1597, pp277-290, 1999.

[LM94] T.V. Lakshman and U. Madhow, “Window-based congestion control for networks with high bandwidth-delay products and random loss: a study of TCP/IP performance”, in Proc. 5th IFIP Conf. On High Performance Networking (HPN ’94), Grenoble, France, June 27-July 1 1994

[LM95a] T.V. Lakshman and U. Madhow, “TCP/IP performance in wide-area networks: the impact of high bandwidth-dealy products and random loss”, in invited paper, Proc. 1995 SIGMETRICS and Performance, p. 262, 1995.

[LM95b] T.V. Lakshman and U. Madhow, “Window-based congestion control in networks with high bandwidth-dealy products and random loss”, in invited paper, Proc. 3rd INFORMS Telecom.Conf. Boca Raton, FL, March 1995.

[LM97] T.V. Lakshman and U. Madhow, “The performance of networks with high bandwidth-dealy products and random loss”, IEEE/ACM Transactions on Networking, June 1997.

[LOR] M. Lorenz & J. Kidd, “Object-Oriented Software Metrics”, Prentice-Hall, 1994.

[MAP] ETSI, “European digital cellular telecommunications system (phase1); Mobile application part specification”, I-ETS 300 044 (GSM 09.02)

[MAZ] S. Mazumdar, N. Mitra, “ROS-to-CORBA Mappings: First Steps towards Intelligent Networking using CORBA”, proceedings of the Conference on Intelligence in Services and Networks, 1997, Como.

[MAZ2] S. Mazumdar, N. Mitra, "Design of a ROS-CORBA Gateway for Inter-operable Intelligent Networking Applications", Bell Labs TM# BL0112540-960930-03TM, June 1997.

[MBJVC00] C. McArdle, R. Brennan, N. Jones, J. Vasic, T. Curran, “Implementation Experience with the OMG IN/CORBA Interworking Specification”, In Proceedings of ICIN, 2000.

[MBO99] A. Misra, J. Baras and T. Ott, “The window distribution of multiple TCPs with random loss queues”, in GLOBECOM ’99, December 1999.

[MCA] C. McArdle & T. Curran, “Optimal Object Placement, Load Distribution and Load Control for Distributed Telecommunication Service Applications” in Proc. 17th International Teletraffic Conference, Brazil, 2001.

[McC00] S. McCreary and K. Claffy, “Trends in wide area IP traffic patterns – a view from Ames internet exchange”, in 13th ITC Specialist Seminar on Internet Traffic Measurement and Modelling, 2000, available at: http://www.caida.org/outreach/papers/

[MGT] 3GPP TS 32.102 3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Telecommunication management; Architecture (Release 6), 2004

[MIT] N. Mitra and S. D. Usiskin, “Interrelationship of the SS7 Protocol Architecture and the OSI Reference Model and Protocols,” The Froehlich/Kent Encyclopedia of Telecommunications, Volume 9, Marcel Dekker, Inc., 1995.

[MM96a] M. Mathis and J. Mahdavi, "Forward Acknowledgment: Refining TCP Congestion Control", Proceedings of SIGCOMM'96, August, 1996, Stanford, CA.

[MM96b] M. Mathis and J. Mahdavi, "TCP Rate-Halving with Bounding Parameters", Technical report. Available from http://www.psc.edu/networking/papers/FACKnotes/current.

[MMFR96] M. Mathis, J. Mahdavi, S. Floyd, A. Romanow, " RFC 2018: TCP Selective Acknowledgement Options", The Internet society, 1996.

[MSMO97] M. Mathis, J. Semke, J. Mahdavi and T. Ott, "The Macroscopic Behaviour of the TCP Congestion Avoidance Algorithm", Computer Communication Review, vol. 27, no. 3, pp. 67-82, July 1997

[NISTNet] M. Carson and D. Santay, "NIST Net: a Linux-based network emulation tool", ACM SIGCOMM Computer Communication Review, vol. 33, no. 3, pp 111-126, July 2003.

[NS] NS project. The Network Simulator—ns-2 (Web site). http://www.isi.edu/nsnam/ns/.

[OB01] P. Olivier and N. Benameur, "Flow level IP traffic characterization", in J. Moreira de Souza, N. L. S da Fonseca and E. A. de Souza e Silva (Eds.) Proceedings of 17th International Teletrafic Congress (ITC-17), pub. Elsevier Science B. V., vol. 4, pp 25-36, Salvador da Bahia, Brazil, September 2001.

[ODP] ITU-T X.901 | ISO/IEC 10746-1 ODP Reference Model Part 1.Overview, ISO, 1995

[OKM96] T. Ott, J. Kemperman, M. Mathis, "The stationary behaviour of ideal TCP congestion avoidance", Bellcore Technical Report, Available at: ftp://ftp.bellcore.com/pub/tjo/TCPWindow.ps, 1996

[OMGIN] R. Brennan (Ed), "CORBA/TC Interworking and SCCP Inter-ORB Protocol Specification", document formal/01-01-01, OMG, January 2001.

[OMGMGT] OMG, "CORBA-TMN Interworking Specification", document formal/2000-08-01, OMG, August 2000.

[OMGSCTP] OMG, "GIOP SCTP Protocol Mapping", Draft Adopted Specification ptc03-08-20, OMG, August 2003.

[OMGWP] OMG, "Intelligent Networking ", white paper: telecom/96-12-03 , December 1996

[OMNI] Ulticom SS.7 Software corporate hompage http://www.ulticom.com/

[Ong99] L. Ong, et al., "RFC 2719: Framework Architecture for Signaling Transport", The Internet Society, 1999.

[Opnet] OpNet corporate homepage http://www.opnet.com

[P508] F. Boujemaa (Ed.), "Introduction of Distributed Computing Middleware in Intelligent Networks (White Paper)", Eurescom, 1997.

[Pad00] J. Padhye, V. Firoiu, D. Townsley, J. Kurose, "Modelling TCP Throughput: A Simple Model and its Empirical Validation", IEEE/ACM Transactions on Networking, vol. 8, no. 2, pp133-145, April 2000. Originally published in ACM SIGCOMM '98, 1998.

[Pad99] J. Padhye, V. Firoiu, D. Townsley, "A stochastic model of TCP reno congestion avoidence and control", Tech. Rep. CMPSCI 99-02, Dept. of Computer Science, University of Massachusetts, Amherst, MA, 1999, Available at: http://www.psc.edu/networking/tcp_friendly.html

[Par03] Parlay Industry Group, ”Parlay 2.1 Specification”, www.parlay.org

[Par98] S. Parker, C. Schmechel, “RFC 2398: Some Testing Tools for TCP Implementors”, The Internet Society, 1998.

[Pax01] V. Paxson and S. Floyd. “Difficulties in Simulating the Internet”. *IEEE/ACM Trans. on Networking*, vol. 9, no. 4, pp392–403, Aug. 2001.

[Pax97a] V. Paxson, “Automated Packet Trace Analysis of TCP Implementations”, Proceedings of ACM SIGCOMM ’97, pp167-179, Cannes, France, August 1997.

[Pax97b] V. Paxson, “Measurements and Analysis of End-to-End Internet Dynamics”, PhD thesis, University of California, Berkley, April 1997.

[Pax99] V. Paxson, et. Al., “RFC 2525: Known TCP implementation problems”, The Internet Society, March 1999. Available at http://www.ietf.org

[PF97] V. Paxson and S. Floyd, “Why We Don’t Know How to Simulate the Internet”, Proceedings 1997 Winter Simulation Conference, Atlanta, December 1997.

[PINT] IETF, “The PINT Service Protocol: Extensions to SIP and SDP for IP Access to Telephone Call Services,” RFC2848, June 2000.

[PTVF92] W. Press, S. Teukolsky, W. Vetterling and B. Flannery, “Numerical Recipes in C: The Art of Scientific Computing”, Cambridge University Press, 1992.

[Q771] ITU-T Rec. Q.771 (1993), Signaling System No. 7 — Functional description of Transaction Capabilities

[Q772] ITU-T Rec. Q.772 (1993), Signaling System No. 7 — Transaction Capabilities information element definitions

[Q773] ITU-T Rec. Q.773, “Signaling System No. 7 — Transaction Capabilities formats and encodings”, 1993

[Q774] ITU-T Rec. Q.774 (1993), Signaling System No. 7 — Transaction Capabilities procedures

[RF99] K. Ramakrishnan and S. Floyd, “A Proposal to add Explicit Congestion Notification (ECN) to IP”, RFC 2481, The Interne

[Rou01] M. Roughan, A. Erramilli and D. Veitch, “Network Performance for TCP Networks Part 1: Persistent Sources”, in Proc. of the International Teletraffic Congress – ITC17, Salvador, Brazil, December 2001.

[SDL92] ITU, Recommendation Z.100, “Specification and Description Language (SDL)”, August 1992

[SLJ01] H.-P. Schwefel, L. Lipsky and M. Jobmann, “On the Necessity of Transient Performance Analysis in Telecommunication Networks”, in J. Moreira de Souza, N. L. S da Fonseca and E. A. de Souza e Silva (Eds.) Proceedings of 17th International Teletrafic Congress (ITC-17), pub. Elsevier Science B. V., vol. 4, pp 1087-1101, Salvador da Bahia, Brazil, September 2001.

[SNACC] R. Joop, "Sample Neufeld Asn.1 to C Compiler", Masters Thesis, 1996. Available at: http://www.fokus.gmd.de/ovma/freeware/snacc/

[SPIR] IETF, SPIRITS charter, http://www.ietf.org/html.charters/spirits-charter.html

[St94] W. Stevens, "TCP/IP Illustrated, vol.1 The Protocols", Addison-Wesley, 1994.

[Ste03] R. Stewart, L. Ong, I. Arias-Rodriguez, K. Poon, P. Conrad, A. Caro, M. Tuexen, "Stream Control Transmission Protocol (SCTP) Implementers Guide", work in progress, available at: http://www.ietf.org/internet-drafts/draft-ietf-tsvwg-sctpimpguide-10.txt, 2003.

[SX00] R. Stewart, Q. Xie, et al., "RFC 2960: Stream Control Transmission Protocol", The Internet Society, 2000.

[T1MGT] ANSI T1, "Working Document for Draft Standard ANSI T1.2xx-2000, "Framework for CORBA-Based Telecommunications Management Network Interfaces"

[TINA] TINA-C, "Overall Concepts and Principals of TINA", Feb 1995. Available at http://www.tinac.com

[TINA97] TINA-C, "Global Convergence of Telecommunications and Distributed Object Computing", IEEE Computer Society, 1998.

[TMF] TMF homepage http://www.tmf.org

[TSAS] OMG, "Telecommunications Service Access and Subscription", document format/2002-12-01, OMG, December 2002.

[Vei01] D. Veitch, "Traffic Modelling, a New Golden Era", Keynote address at ITC17, Salvador, Brazil, December 2001. Slides available at: http://www.cubinlab.ee.mu.oz.au/~darryl/ITC_keynote.pdf

[VISI] Visibroker product homepage http://www.visibroker.com

[WM01] J. Handley and M. Handley, "Extending equation-based congestion control to multicast applications", in ACM SIGCOMM, August 2001.

[X500] ITU-T Rec. X.519, "Information Technology – Open Systems Interconnection — The Directory: Protocol Specifications", 1993

[X680] ITU-T Rec. X.680 (1994), Information Technology — Open Systems Interconnection – Abstract Syntax Notation One (ASN.1): Specification of basic notation.

[X880] ITU-T Rec. X.880 (1994), Information Technology — Remote Operations: Concepts, model and notation

[XS02] S. Xu, T. Saadawi, "Revealing the Problems with 802.11 MAC Protocol in Multi-hop Wireless Ad Hoc Networks", Journal of Computer Networks. Vol. 38, No. 4, March 2002

[YSL02] G. Ye, T. Saadawi, M. Lee, "SCTP Congestion Control Performance in Wireless Multi-hop Networks", MILCOM'02, October 2002.

[Gay96] M. Gaynor, "Proactive Packet Dropping Methods for IP Gateways", http://www.eecs.harvard.edu/~gaynor/final.ps, 1996

List of Acronyms

3GPP: 3rd Generation Partnership Project
ACM: Association for Computing Machinery
AIN: Advanced Intelligent Network
ANSI: American National Standards Institute
AQM: Active Queue Management
ASN.1: Abstract Syntax Notation One
ATM: Asynchronous Transfer Mode
BER: Basic Encoding Rules
CAMEL: Customised Access for Mobile network Enhanced Logic
CCF: Call Control Function
CCS: Common Channel Signalling
cdf: cumulative distribution function
CORBA: Common Object Request Broker Architecture
CPE: Customer Premises Equipment
CS-1: Capability Set 1
CSL: Component Sub-Layer
DCE: Distributed Computing Environment
DCOM: Distributed Common Object Model
DIN: Distributed Intelligent Network
DPC: Destination Point Code
DPE: Distributed Processing Environment
ECN: Explicit Congestion Notification
ESIOP: Environment Specific Inter-Orb Protocol
ETSI: European Telecommunications Standardisation Institute
FE: Functional Entity
FIFO: First-In-First-Out
FR: Fast Retransmit
FTP: File Transfer Protocol
GIOP: General Inter-Orb Protocol
GSM: Global System for Mobile communication
GTA: Global Title Address
GTT: Global Title Translation
HTTP: Hyper-Text Transfer Protocol
IAB: Internet Architecture Board
IETF: Internet Engineering Task Force
IDL: Interface Definition Language
IF: Information Flow

IN: Intelligent Network
INAP: Intelligent Network Application Protocol
IOP: Inter-Orb Protocol
IP: Intelligent Peripheral *or* Internet Protocol
IPC: Inter-Process Communication
ISDN: Integrated Services Digital Network
ISO: International Standards Organisation
IT: Information Technology
ITU: International Telecommunications Union
LRD: Long Range Dependence
JIDM: Joint Inter-Domain Management
MAP: Mobile Application Part
MoIP: Multimedia over Internet Protocol
MPLS: Multi-Protocol Label Switching
MSC: Mobile Switching Centre
MSS: Maximum Segment Size
MSU: Message Signal Unit
MTP: Message Transfer Part
MTP1: Message Transfer Part – Level 1
MTP2: Message Transfer Part – Level 2
MTP3: Message Transfer Part – Level 3
MTU: Maximum Transmission Unit
NS: Network Simulator
ODP: Open Distributed Processing
OMG: Object Management Group
OPC: Origination Point Code
ORB: Object Request Broker
OSI: Open Systems Interconnection
PBX: Private Branch eXchange
PDA: Personal Digital Assistant
PC: Point code *or* Personal Computer
pdf: probability distribution function
PFTK: Padhye, Firoiu, Towsley and Kurose
PINT: PSTN/IN Interworking
PLMN: Public Mobile Land Network
pmf: probability mass function
POTS: Plain Old Telephony System
PSTN: Public Switched Telephone Network
RED: Random Early Detection
ROS: Remote Operations Service
RSVP: Resource reSerVation Protocol
RTO: Retransmission Time Out
RTT: Round Trip Time

SACK: Selective ACKnowledgement
SCCP: Signalling Connection Control Part
SCEF: Service Creation Environment Function
SCF: Service Control Function
SCP: Service Control Point
SCTP: Stream Control Transmission Protocol
SDF: Service Data Function
SDP: Service Data Point
SECIOP: SECure Inter-Orb Protocol
SIP: Session Initiation Protocol
SLA: Service Level Agreement
SLP: Service Logic Program
SMAF: Service Management Agent Function
SME: Small to Medium Enterprise
SMF: Service Management Function
SOAP: Simple Object Access Protocol
SP: Signalling Point
SPIRITS: Service in the PSTN/IN requesting Internet Service
SRF: Specialised Resource Function
SS.7: Signalling System No.7
SSCP: Service Switching Control Point
SSF: Service Switching Function
SSN: Sub-System Number *or* Stream Sequence Number
SSP: Service Switching Point
STP: Signalling Transfer Point
TC: Transaction Capabilities
TD: Triple Duplicate
TCAP: Transaction Capabilities Application Part
TCP: Transmission Control Protocol
TFC: Transfer Controlled
TINA: Telecommunications Information Networking Architecture
TIPHON: Telecommunications and Internet Protocol Harmonisations Over Networks
TMF: Tele-Management Forum
TSL: Transaction Sub-Layer
TSN: Transmission Sequence Number
UDP: User Datagram Protocol
UML: Unified Modelling Language
VoIP: Voice over Internet Protocol
VPN: Virtual Private Network
W3C: World Wide Web Consortium
WAP: Wireless Access Protocol
XML: eXtensible Markup Language

Appendix 1

TC-User ASN.1 Specification to OMG IDL Translation Algorithm

The interfaces of TC-User applications are described in ASN.1 modules. To convert these to descriptions of CORBA entities, these modules must be translated into IDL specifications. There are three main forms of description:

4. One using ASN.1 and natural language (for example ITU-T INAP CS-1)
5. One using just ASN.1 (for example ETSI INAP CS-1). In this case additional ASN.1 macros are used to formally define the parts of the specification; otherwise, defined using natural language.
6. Newer specifications (for example ITU-T CS-2) may be identified as a third type as they use the revised ASN.1 notation (information objects instead of macros).

All three forms of specifications are considered in the mappings provided below.

Type I ASN.1 Description of TC-User

An older ASN.1 and natural language description (henceforth called a Type I TC-User description) has the general form:

```
MODULE ::= <module_name>
BEGIN
IMPORTS <import_list>
EXPORTS <export_list>
OPERATION MACRO_1
...
OPERATION MACRO_n
ERROR MACRO_1
...
ERROR MACRO_m
END
```

Figure A1.1: ASN.1 form of Type I Specification

Type II ASN.1 description of TC-User

An older ASN.1 only description (henceforth called a Type II TC-User description) has the general

form:

```
MODULE ::= <module_name>
BEGIN
IMPORTS <import_list>
EXPORTS <export_list>
OPERATION MACRO_1
...
OPERATION MACRO_n
ERROR MACRO_1
...
ERROR MACRO_m
APPLICATION-SERVICE-ELEMENT MACRO_1
...
APPLICATION-SERVICE-ELEMENT MACRO_x
APPLICATION-CONTEXT MACRO_1
...
APPLICATION-CONTEXT MACRO_y
END
```

Figure A1.2: ASN.1 form of Type II Specification

Type III ASN.1 description of TC-User

A new (post 1992) ASN.1 only description (henceforth called a Type III TC-User description) has the general form:

```
MODULE ::= <module_name>
BEGIN
IMPORTS <import_list>
EXPORTS <export_list>
OPERATION information object_1
...
OPERATION information object_n
ERROR information object_1
...
ERROR information object_m
OPERATION-PACKAGE information object_1
...
OPERATION-PACKAGE information object_x
CONTRACT information object_1
...
CONTRACT information object_y
END
```

Figure A1.3: ASN.1 form of Type III Specification

Mapping Algorithm

The JIDM Specification Translation document does not define algorithms for converting any of the ASN.1 macros or equivalent information objects used in the above specifications. This is the primary extension to the JIDM specification presented in this document. A mapping is also defined for the

ASN.1 EXTENSION macro/information object as it is extensively used in several TC-user OPERATION definitions.

In general:

- An instance of an OPERATION provides zero or one RESULT definition and refers to zero, one, or more instances of an ERROR information object class/macro.
- An instance of an APPLICATION-SERVICE-ELEMENT/OPERATION-PACKAGE refers to one or more OPERATIONs.
- An instance of an APPLICATION-CONTEXT/CONTRACT refers to one or more instances of an APPLICATION-SERVICE-ELEMENT/OPERATION-PACKAGE.

The basic scheme for the static translation of a TC/ROS-User specification to a CORBA IDL specification is as follows:

1. Map a TC-User application definition contained within an ASN.1 module into an IDL module called **<module_name>** in accordance with the standard JIDM ASN.1 module to IDL module name mapping rules and

- all interfaces, types, and constants generated from an ASN.1 module will be within the scope of the corresponding IDL module
- declare the imported types in the ASN.1 module as **typedef**s of imported IDL types
- declare an IDL interface, called **TcUserFactory** that inherits from the interface **TcSignaling::TcUserGenericFactory**, if the current TC-User description is Type I and there is at least one instance of a ROS/TC OPERATION defined, or if the current TC-User description is Type II/III and there is at least one instance of an APPLICATION-CONTEXT/CONTRACT defined.

2. Map each of the ASN.1 types into a corresponding IDL type using the translation scheme defined by the JIDM. A complex ASN.1 data type (used to describe TC-User PDUs) may generate more than one IDL data type.
3. For a Type I description module that has a **TcUserFactory** interface defined:

- Create an interface called **DefAc**.
- Create an IDL exception definition (as described in Section “Mapping for ERRORs,”) for every ERROR instance declared or imported into the module.
- Create an IDL **return** type in the operation signature (as described in Section “Mapping for OPERATIONs,”) for every OPERATION instance with a RESULT keyword declared or imported into the module.
- Create an IDL operation definition within **DefAc** (as described in Section, “Mapping for

OPERATIONs,") for every OPERATION instance declared or imported into the module.

- Define an operation (**create_DefAcInitiator**) within the scope of the **TcUserFactory** interface. This operation will return an object of type **<module_name>::DefAc**. This operation has no parameters.
- A **raises** clause containing the exceptions: TcSignaling::NoMoreAssociations, TcSignaling::UnsupportedTcContext - define an operation (**create_DefAcResponder**) within the scope of the **TcUserFactory** interface. This operation will return an object of type **<module_name>::DefAc**. This operation has the following parameters:
  - an in parameter of type **<module_name>::DefAc** called initiator
  - an in parameter of type **TcSignaling::AssociationId** called **a_id**
  - an in parameter of type **TcSignaling::TcContextSetting** called **tc_context_setting**
  - a **raises** clause containing the exceptions **TcSignaling::NoMoreAssociations, TcSignaling::UnsupportedTcContext**
- • Define[4] an operation (**create_DefAcResponder_with_dialogdata**) within the scope of the **TcUserFactory** interface. This operation will return an object of type **<module_name>::DefAc**. This operation has the following parameters:
  - an in parameter of type **<module_name>::DefAc** called **initiator**
  - an in parameter of type **TcSignaling::AssociationId** called **a_id**
  - an in parameter of type **TcSignaling::TcContextSetting** called **tc_context_setting**
  - an in parameter of type string called **protocol_version**
  - an in parameter of type **TcSignaling::DialogUserData** called **d_u_d**
  - a **raises** clause containing the exceptions **TcSignaling::NoMoreAssociations**, **TcSignaling::InvalidParameter**, and **TcSignaling::UnsupportedTcContext**

4. For a Type II/III description module that has a **TcUserFactory** interface defined:

- Create an IDL exception definition (as described in Section, "Mapping for ERRORs,") for every ERROR instance declared or imported into the module.
- Create an IDL return type for the operation signature (as described in Section, "Mapping for OPERATIONs,") for every OPERATION instance with a RESULT keyword declared or imported into the module.
- Create two interfaces: **<contract_name>Initiator** and **<contract_name>Responder** (as described in Section, "CONTRACT (or APPLICATION-CONTEXT) Mapping,") for every

[4] . It is unlikely that a Type 1 TC-User description uses the TC DialogControl portion. This create operation was put in for completeness to cover this unlikely case. The authors have found no example of such use.

APPLICATION-CONTEXT/CONTRACT instance.

- Define an operation (**create_<contract_name>Initiator**) within the scope of the **TcUserFactory** interface. This operation will return an object of type **<module_name>::<contract_name>Initiator**. This operation has no parameters.
- Define an operation (**create_<contract_name>Responder**) within the scope of the **TcUserFactory** interface. This operation will return an object of type **<module_name>::<contract_name>Responder**. This operation has the following parameters:
    - an in parameter of type **<module_name>::<contract_name>Initiator** called **initiator**
    - an in parameter of type **TcSignaling::AssociationId** called **a_id**
    - an in parameter of type **TcSignaling::TcContextSetting** called **tc_context_setting**
    - a **raises** clause containing the exceptions: **TcSignaling::NoMoreAssociations**, **TcSignaling::UnsupportedTcContext**;
- Define an operation (**create_<contract_name>Responder_with_dialogdata**) within the scope of the **TcUserFactory** interface. This operation will return an object of type **<module_name>::<contract_name>Responder**. This operation has the following parameters:
    - an in parameter of type **<module_name>::<contract_name>Initiator** called **initiator**
    - an in parameter of type **TcSignaling::AssociationId** called **a_id**
    - an in parameter of type **TcSignaling::TcContextSetting** called **tc_context_setting**
    - an in parameter of type **string** called **protocol_version**
    - an in parameter of type **TcSignaling::DialogUserData** called **d_u_d**
    - a **raises** clause with the exceptions **TcSignaling::NoMoreAssociations**, **TcSignaling::UnsupportedTcContext**, and **TcSignaling::InvalidParameter**

5. The compiler will generate data for an operation code repository that will provide access via the interface **TcRepository** to the following elements: IDL Scoped name for the ASN.1 operation or error, IdType (i.e., local (integer) or global (ASN.1object identifier)), ID (i.e., the actual value defined in the ASN.1 specifications), Type (i.e., operation or error), Timer (associated with the operation).

Generation of TC Repository to hold ScopedName to ID Mapping

In addition to the IDL file produced during specification translation, a TC Repository is created that contains the information about mapping the IDL scoped name for interfaces and identifier constants to a corresponding TC operation, error and, if present, extension data type identifiers. This information is used to match ROS/TC operation codes, extensions, and error codes to specific IDL constructs.

The **TcRepository** interface provides access to the following information:

- *IDL scoped name of the IDL construct*: this represents the unique name in the Interface Repository of the corresponding ASN.1 module element (operations, errors and, if present, extension data types).
- *the type of identifier*: this allows constructs (usually operations or errors) to be named by either a local value or an object identifier.
- *the identifier value*: the operation or error code (or other identifier) is placed here.
- *the timer value*: this is the maximum length of an operation timer in seconds. If no timer value is formally defined for an operation then a value of 0 is used.

An example of the sort of information captured in such a repository is shown in figure A1.4, where Q1218_3 is the module name of an IN-specific ASN.1 module converted to IDL using specification translation.

```
# Scoped-Name                                IdType ID Type      Timer
Q1218_3::DefAc::initialDP                     local 0  operation  0
Q1218_3::DefAc::originationAttemptAuthorized local 1  operation  0
Q1218_3::DefAc::canceled                      local 0  error      None
```

Figure A1.4: Example TC Repository Contents

Mapping of TC-User/ROS Constructs

This section defines detailed mapping rules for the individual ASN.1 constructs required for TC/ROS Specification Translation. Examples of all mappings are also provided.

Mapping for OPERATIONs

The OPERATION information object class (or MACRO in the case of Type I and II descriptions) is used to define ROS/TC-User operations within the scope of a ROS/TCUser protocol. Each instance of a ROS/TC-User OPERATION is mapped to an IDL operation signature. The mapping is defined below.

Operation Instance Mapping

1. Map the ROS operation name to an IDL operation name according to the JIDM type name mapping rules.

2. Map the keyword ARGUMENT (which is a single ASN.1 type) to a single in argument type of the IDL operation. The ASN.1 ARGUMENT type is mapped to an IDL type using the JIDM specification translation rules.
3. If the operation has no LINKED keyword, create an **inout** argument of type **TcContext** called **ctext**. If the operation has a LINKED keyword, create an **inout** argument of type **TcLinkedContext** called **lctext**. This will be used to carry TC dialog handling information, invoke ids, association ids, and linked ids for operations that may have linked replies.
4. Map the ROS RESULT argument (which is a single ASN.1 type) to the IDL result type in the operation signature. The ASN.1 RESULT type is mapped to an IDL return type using JIDM specification translation rules. If there is no ASN.1 data type following the RESULT keyword or the RESULT keyword is absent, the IDL result type is a **void**.
5. If there are ERRORS defined for an operation, then add an IDL raises expression to the operation signature. For each ASN.1 ERROR type following the ERRORS keyword add its translated (see Section, "Mapping for ERRORs,") IDL exception name to the list in the **raises** clause.

Operation Identifier Assignment Mapping

The operation code assignment is used to create the TC Repository as described in Section, "Generation of TC Repository to hold ScopedName to ID Mapping".

OPERATION Mapping Example

The example below shows how an **OPERATION** instance may be defined for a TCUser, in this case an operation defined by IN Capability Set 1 (CS-1). The argument of the operation is of type **InitialDPArg**. The operation only reports unsuccessful completion by one set of errors.

```
InitialDP ::= OPERATION
ARGUMENT InitialDPArg
ERRORS { MissingCustomerRecord, MissingParameter, SystemFailure,
TaskRefused, UnexpectedComponentSequence, UnexpectedData-
Value, UnexpectedParameter }
```

Figure A1.5: The InitialDP ASN.1 Operation Definition

The mapping of the ASN.1 operation to an IDL operation is shown in figure A1.6 below:

```
void InitialDP (in InitialDPArgType InitialDPArg, inout TcContext ctext)
raises (MissingCustomerRecord, MissingParameter, SystemFailure, TaskRefused,
UnexpectComponentSequence, UnexpectedDataValue, UnexpectedParameter);
```

Figure A1.6: The Mapped IDL of the InitialDP Operation

The ASN.1 operation argument type **InitialDPArg** is mapped to the IDL type **InitialDPArgType**

and an **in** parameter of this type is placed in the IDL operation parameter list. The IDL operation also takes an **inout** parameter of type **TcContext**, which is used to carry dialog flow control information, the operation's invokeID and the AssociationId as context information when the operation is invoked. The ASN.1 errors are mapped to parameters of the **raises** clause in the IDL operation signature. As the operation does not have a result returned, it is necessary to insert the IDL result type **void**.

Mapping for ERRORs

Each instance of a ROS ERROR information object class (or macro, for Type I and II descriptions) is mapped to an IDL user defined exception. The mapping is defined below. Appendix E provides the ASN.1 definition for the ERROR information object class as well as the macro.

Error Instance Mapping

1. Map the ROS **ERROR** instance's name to an IDL exception name.
2. If there is a **PARAMETER** defined as an ASN.1 type for the error instance, map it to a single parameter of an IDL exception using the JIDM translation rules.
3. If there is a **PARAMETER** defined as an unnamed ASN.1 construct, map the parameter of the error to a single IDL type with name **<ErrorName>Param** and use the JIDM translation rules for the contents of this type. This type is defined as a single parameter of the IDL exception created.
4. Create an additional parameter in the IDL exception of type **DialogFlowCtr** with name **ctr**.

Error Identifier Assignment Mapping

This form of the macro is used to create an entry in the TC Repository as described in the Section, "Generation of TC Repository to hold ScopedName to ID Mapping".

ERROR mapping example

A mapping for an instance of an error taken from IN CS-1 is shown below. The ASN.1 for an error instance named SystemFailure which conveys a single parameter UnavailableNetworkResource is as follows:

```
SystemFailure ::= ERROR
PARAMETER UnavailableNetworkResource
```

Figure A1.7: The ASN.1 of the SystemFailure Error

It maps to the IDL shown in figure A1.8.

```
exception SystemFailure (UnavailableNetworkResourceType
unavailableNetworkResource; DialogFlowCtr ctr;);
```

Figure A1.8: The Mapped IDL of the SystemFailure Error

Mapping for the EXTENSION MACRO

The ASN.1 EXTENSION MACRO is defined in ITU-T Recommendation Q.1400 for the purpose of allowing extensions to be made to standardized TC application protocols. The use of the **EXTENSION** MACRO is intended only for minor extensions to an abstract syntax, for example to allow the addition of information elements that may enhance an activity but are not essential to performing the basic activity or to allow the addition of a capability that is not essential to the base capability. All data types defined in Interface Recommendation for IN CS-1 use this extension mechanism and so an ASN.1-to-IDL mapping of the **EXTENSION** MACRO is required to fully support standard application protocols (e.g., INAP) at a gateway. The **EXTENSION** MACRO is defined in figure A1.9 below.

```
EXTENSION MACRO::=
TYPE NOTATION::= ExtensionType Criticality
VALUE NOTATION::= value (VALUE CHOICE {
private-extension INTEGER,
standard-extension OBJECT IDENTIFIER })
ExtensionType::= EXTENSION-SYNTAX type | empty
Criticality::= CRITICALITY value (Criticality Type)
CriticalityType::= ENUMERATED { ignore(0), abort(1) }
```

Figure A1.9: The ASN.1 of the EXTENSION MACRO

EXTENSION instance mapping

Every **EXTENSION** Macro instance will result in the creation of two IDL types, an extension criticality identifier and an extension type.

1. The instance of the **EXTENSION** macro **EXTENSION SYNTAX** field is mapped to an IDL type according to the standard JIDM ASN.1 type mapping rules. If the ASN.1 Type is named, the standard JIDM mapping to an IDL type is performed, if unnamed it is given the same name as the extension (barring the need for JIDM disambiguation).

2. The **CRITICALITY** field of the macro is mapped to an IDL constant declaration of the **ASN1_ExtensionCriticality** type with an identifier equal to the extension name appended with the string "ExtensionCriticality" and an appropriate value. If there is no **CRITICALITY** declaration, an IDL type with criticality **ASN1_EXTENSION_ABORT** is declared as this is the default.

3. The macro VALUE declaration and the **ScopedName** of the IDL Extension Type is placed in the TC Repository.

EXTENSION mapping example

The following example in figure A1.10 is taken from ETSI INAP CS-1 (where it is used as an example of the **EXTENSION** macro).

```
SomeNetworkSpecificIndicator ::= EXTENSION
EXTENSION SYNTAX BOOLEAN
CRITICALITY abort
someNetworkSpecificIndicator SomeNetworkSpecificIndicator ::= 1
```

Figure A1.10: The ASN.1 of the SomeNetworkSpecificIndicator EXTENSION

The IDL this maps to is given in figure A1.11 below.

```
typedef ASN1_Boolean SomeNetworkSpecificIndicator;
const ASN1_ExtensionCriticality
SomeNetworkSpecificIndicatorExtensionCriticality = ASN1_EXTENSION_ABORT;
```

Figure A1.11: Mapped IDL of the SomeNetworkSpecificIndicator EXTENSION

CONTRACT (or APPLICATION-CONTEXT) Mapping

The **CONTRACT** information object class (or **APPLICATION-CONTEXT** macro in the case of Type II descriptions) is used to define the rules of engagement for a ROS/TCUser protocol. Each instance of a **ROS CONTRACT** is mapped to an IDL interface. The mapping is defined below.

Contract (or Application Context) Instance Mapping

When mapping a Type III (respectively Type II description), an instance of a **CONTRACT** (respectively **APPLICATION-CONTEXT** macro) is used to create IDL interface definitions as follows:

1. Create one interface named **<contract_name>Responder**, and within this interface create IDL operation signatures (as described in Section, "Operation Instance Mapping,") for:

- each ROS operation identified by the keyword **CONSUMER INVOKES** for each **OPERATION-PACKAGE** (or ASE) identified by the keyword **INITIATOR CONSUMER OF** for the **CONTRACT/APPLICATION-CONTEXT** instance definition.
- each ROS operation identified by the keyword **SUPPLIER INVOKES** for each **OPERATION-PACKAGE** (or ASE) identified by the keyword **RESPONDER CONSUMER OF** for the **CONTRACT/APPLICATION-CONTEXT** instance definition.
- each ROS operation identified by the keyword **OPERATIONS** for each **OPERATION-PACKAGE** (or ASE) identified by either of the keywords **INITIATOR CONSUMER OF** or **RESPONDER CONSUMER OF** for the **CONTRACT / APPLICATION-CONTEXT** instance definition.
- each ROS operation identified by the keywords **OPERATIONS**, **CONSUMER INVOKES**, and **SUPPLIER INVOKES** for each **OPERATION-PACKAGE** (or ASE) identified by the keyword

OPERATIONS OF for the **CONTRACT/APPLICATION-CONTEXT** instance definition.

2. Create one interface named **<contract_name>Initiator**, and within this interface create IDL operation signatures (as described in Section , "Operation Instance Mapping,") for:

- each ROS operation identified by the keyword **CONSUMER INVOKES** for each **OPERATION-PACKAGE** (or ASE) identified by the keyword **RESPONDER CONSUMER OF** for the **CONTRACT/APPLICATION-CONTEXT** instance definition.
- each ROS operation identified by the keyword **SUPPLIER INVOKES** for each **OPERATION-PACKAGE** (or ASE) identified by the keyword **INITIATOR CONSUMER OF** for the **CONTRACT/APPLICATION-CONTEXT** instance definition.
- each ROS operation identified by the keyword **OPERATIONS** for each **OPERATION-PACKAGE** (or ASE) identified by either of the keywords **INITIATOR CONSUMER OF** or **RESPONDER CONSUMER OF** for the **CONTRACT / APPLICATION-CONTEXT** instance definition.
- each ROS operation identified by the keywords **OPERATIONS**, **CONSUMER INVOKES**, and **SUPPLIER INVOKES** for each **OPERATION-PACKAGE** (or ASE) identified by the keyword **OPERATIONS OF** for the **CONTRACT/APPLICATION-CONTEXT** instance definition.

3. Define an operation (**create_<interface_name>Responder**) within the scope of the **TcUserFactory** interface for each of the above-mentioned responder interfaces. Refer to the earlier Section, "Mapping Algorithm," (item 4) for more information.
4. Define an operation (**create_<interface_name>Responder_with_dialogdata**) within the scope of the **TcUserFactory** interface for each of the above-mentioned responder interfaces. Refer to the earlier Section, "Mapping Algorithm," (item 4) for more information.
5. Define an operation (**create_<interface_name>Initiator**) within the scope of the **TcUserFactory** interface for each of the above-mentioned initiator interfaces. Refer to the earlier Section, "Mapping Algorithm," (item 4) for more information.

Contract (or Application Context) Mapping Example

The example in figure A1.12 below shows how a ROS/TC Contract is defined in ASN.1. The Application Context instance **exampleContext1** includes the **OPERATIONPACKAGE** instance **examplePackage1**, which is also defined below. The dialog initiator may invoke operations **exampleOp1** and **exampleOp2**. The dialog responder may not invoke any operation.

```
exampleContext1 CONTRACT ::=
{
INITIATOR CONSUMER OF {examplePackage1}
ID objectIdentifierOfexampleContext1
}

examplePackage1 OPERATION-PACKAGE ::=
{
CONSUMER INVOKES {exampleOp1 | exampleOp2}
ID objectIdentifierOfexamplePackage1
}
```

Figure A1.11: The ASN.1 Contract exampleContext1

The mapping of the Contract to an IDL interface is as shown in the following figure A1.12.

```
interface exampleContext1Initiator {
<resultType> exampleOp1(<params>);
<resultType> exampleOp2(<params>);
};
```

Figure A1.12: The IDL Mapping of Contract exampleContext1

Only one interface, **exampleContext1Initiator**, is generated in this case. Within this interface two operation signatures are generated. These operations correspond to the two ROS operations listed in the **CONSUMER INVOKES** clause of the **examplePackage1** operation package, which is then listed in the **INITIATOR CONSUMER OF** clause of the **exampleContract1** contract definition. There is no responder interface generated for this example as only the initiator is a consumer of the **examplePackage1** operation package and the operation package definition specifies that only the **CONSUMER INVOKES**.