Netztechnische Aufgaben in VIOLA
T2.5: Netztechnische Untersuchungen und Tests – VIOLA-Labor
Bericht B2.5.6

Studie zur Skalierbarkeit der eingesetzten Netztechniken

besteht aus:

- **Skalierbarkeit des Netzwerkreservierungssystems ARGON**
- Master Thesis (Sajid Rashid Khan):
  *Deployment and evaluation of GMPLS Networks in Context of Scalability.*

31.01.2007

Enthält Beiträge der Konsortialpartner:

Universität Bonn
FhG.IAIS
Skalierbarkeit des Netzwerkreservierungssystems ARGON

25.01.2007

Universität Bonn

Uli Bornhauser
Christoph Barz
Wolfgang Moll
Markus Pilz

Das diesem Bericht zugrunde liegende Vorhaben wurde mit Mitteln des Bundesministeriums für Bildung und Forschung unter dem Förderkennzeichen 01AK605J gefördert. Die Verantwortung für den Inhalt dieser Veröffentlichung liegt bei den Autoren.
Das Institut für Informatik IV ist ein Institut der Rheinischen Friedrich-Wilhelms-Universität Bonn. Es wird durch Prof. Dr. Peter Martini vertreten.

Zuständige Aufsichtsbehörde:
Ministerium für Wissenschaft und Forschung des Landes Nordrhein Westfalen,
Völklinger Strasse 49,
40221 Düsseldorf.

Das diesem Bericht zugrunde liegende Vorhaben wurde mit Mitteln des Bundesministeriums für Bildung und Forschung unter dem Förderkennzeichen 01AK605J gefördert. Die Verantwortung für den Inhalt dieser Veröffentlichung liegt bei den Autoren.
Inhalt

1 Einleitung ................................................................................................................... ........ 4

1.1 Komponenten im ARGON System ............................................................................ 4

1.1.1 Kernkomponenten .............................................................................................. 4

1.1.2 Subkomponenten ................................................................................................ 9

1.2 Zusammenwirken der Einzelkomponenten .............................................................. 11

1.2.1 Parallelisierung der Anfragebearbeitung ......................................................... 11

2 ARGON Eingabeparameter .............................................................................................. 13

2.1 Fehlertoleranz ........................................................................................................... 13

2.1.1 Ausfall einzelner Netzkomponenten ................................................................ 13

2.1.2 Fehler im Reservierungssystem ....................................................................... 14

2.2 Laufzeiteinteile zur Bearbeitung von Reservierungsanfragen .................................. 14

2.3 Netztopologien ......................................................................................................... 14

2.3.1 Preprocessing ................................................................................................... 15

2.3.2 Path Computation ............................................................................................. 15

2.3.3 Postprocessing .................................................................................................. 16

2.4 Anzahl angenommener Reservierungsanfragen ....................................................... 17

2.4.1 Preprocessing ................................................................................................... 17

2.4.2 Path Computation ............................................................................................. 22

2.4.3 Postprocessing .................................................................................................. 23

2.5 Parameter einer Reservierungsanfrage ..................................................................... 23

2.5.1 Preprocessing ................................................................................................... 23

2.5.2 Path Computation ............................................................................................. 24

2.5.3 Postprocessing .................................................................................................. 24

2.5.4 Signalisierung ................................................................................................... 24

3 Ausblick ..................................................................................................................... ...... 26

3.1 Zeitslotbasierte Sperrung des Ressource Managers ................................................. 26

3.2 Reservierungen ohne Ende-Zu-Ende Verzögerungsanforderungen ......................... 26

3.2.1 Hierarchisierung des Reservierungssystems .................................................... 27

3.2.2 Ansätze zur Erhöhung der Fehlerresistenz ....................................................... 27

4 Anhang ....................................................................................................................... ...... 29

4.1 Referenzen ............................................................................................................... 29

4.2 Akronyme ................................................................................................................. 29
1 Einleitung

Das ARGON Reservierungssystem bildet die Kernkomponente, um Reservierungen innerhalb des VIOLA Netzes umzusetzen. Anfragen, die an das System gestellt werden, werden hierbei durch verschiedene Subkomponenten bearbeitet. Durch welche Komponente eine Anfrage bearbeitet wird, hängt von ihrem Typ und den spezifizierten Parametern ab. Als Anfragetypen werden Reservierungs-, Änderungs-, Verfügbarkeits-, Lösch-, Bind-, Aktivierungs- und Abbfrageanfragen durch das System unterschieden. Eine genaue Spezifikation der Anfragen findet sich in [[B2.3.2].

Im Folgenden sollen zunächst die verschiedenen Komponenten im ARGON System skizziert und deren Zusammenarbeit im Hinblick auf die Bearbeitung einer Reservierungsanfrage erläutert werden. Aufbauend auf diesen Betrachtungen sollen im Folgenden eine Reihe von Aspekten untersucht werden, welche die Skalierbarkeit des Systems in Hinblick auf wachsende Topologiegrößen und steigende Anfragezahlen betreffen. Zusätzlich dazu soll der Einfluss einiger Eingabeparameter auf die Bearbeitungszeit von Reservierungsanfragen skizziert werden.

Ausgehend von diesen Untersuchungen sollen Empfehlungen zur Weiterentwicklung der in ARGON verwendeten Algorithmen und Datenstrukturen gegeben werden.

1.1 Komponenten im ARGON System

Eine Anfrage an das ARGON Reservierungssystem muss eine Reihe von Komponenten durchlaufen, um vollständig durch das System bearbeitet zu werden. Die modulare Einteilung in Komponenten erlaubt hierbei im Allgemeinen eine verzahnte Bearbeitung der Anfragen. Je nach Anfragetyp ist sogar eine vollständig parallele Bearbeitung der Anfragen, die an das System gestellt werden, möglich.

In diesem Abschnitt sollen zunächst die Kern- und Subkomponenten des Systems vorgestellt und deren grundlegende Arbeitsweise erläutert werden. Diese beruhen bereits auf den Empfehlungen für das Redesign des Reservierungssystems. Die Zusammenarbeit der einzelnen Komponenten zur Bearbeitung der verschiedenen Anfragen wird dann in Abschnitt 1.2 genauer beschrieben.

1.1.1 Kernkomponenten

Die sogenannten Kernkomponenten des Systems haben die Aufgabe, die Bearbeitung der einzelnen Anfragen an das System zu koordinieren. Die eigentliche Bearbeitung der Anfragen wird durch die so genannten Subkomponenten, welche im Abschnitt 1.1.2 genauer erläutert sind, durchgeführt.

Im Network Resource Provisioning System (NRPS) ARGON werden zwei Arten von Anfragen, die parallelisierbare und die nicht-parallelisierbare Anfragen, unterschieden. Während parallelisierbare Anfragen durch das Reservierungssystem parallel angearbeitet werden, werden nicht-parallelisierbare Anfragen durch das Reservierungssystem in einigen wichtigen Schritten seriell abgearbeitet. Zu den nicht-parallelisierbaren Anfragen zählen Reservierungs-, Änderungs- und Verfügbarkeitsanfragen.

Nicht-parallelisierbare Anfragen haben die Eigenschaft, dass das Ergebnis einer solchen Anfrage im Allgemeinen Einfluss auf das Ergebnis folgender, nicht-parallelisierbarer Anfragen hat. Eine vollständig parallelisierte Bearbeitung dieser Anfragen kann hierbei – soweit nicht weitere Locking-Mechanismen auf den Datenstrukturen implementiert werden – zu Inkon sistenzen wie einer mehrfachen Einplanung der gleichen Ressourcen führen. Im Extremfall könnte dies zu Situationen führen, in denen eine Anfrage zwar durch das Reservierungssys-
Skalierbarkeit des Netzwerkreservierungssystems ARGON

tem angenommen wurde, diese zum Signalisierungszeitpunkt im Netz wegen mangelnder Ressourcen aber nicht umsetzbar ist. Grundsätzlich ist aber eine verzahnte Art der Anfragebearbeitung möglich. Das Reservierungssystem sorgt dabei für eine serielle Bearbeitung der kritischen Operationen wie beispielsweise der Pfadberechnung. Nicht-kritische Operationen wie die Validierung einer Anfrage oder die Generierung der Antwort werden weiterhin parallel abgearbeitet.

Im Gegensatz zu diesem Typ von Anfragen werden parallelisierbare Anfragen durch das System unabhängig von anderen Anfragen vollständig parallel bearbeitet. Hierzu zählen Bind-, Aktivierungs-, Lösch-, und Abfrageanfragen. Eine Bearbeitung dieser Anfragen kann nicht zu einer mehrfachen Einplanung derselben Ressourcen führen.

1.1.1.1 Request Dispatcher

Der Request Dispatcher realisiert die interne Schnittstelle des ARGON Reservierungssystems zur Annahme von Reservierungen. Des Weiteren werden durch ihn einige grundlegende Operationen durchgeführt, die für die weitere Bearbeitung von Anfragen beliebigen Typs notwendig sind. Im Folgenden wird die Anfrage an die Kernkomponente des System weitergeleitet, welche für die weitere Bearbeitung der Anfrage zuständig ist.

Anfragen an das System werden mittels eines Java Message Service (JMS) Topic, im weiteren Dokument als IRes Schnittstelle bezeichnet, entgegengenommen. Mehrere parallele Instanzen des Request Dispatchers nehmen an dieser Schnittstelle eingehende Anfragen entgegen.

Die IRes Schnittstelle dient nicht als externe Schnittstelle für Anwender. Vielmehr stellt diese eine Schnittstelle zwischen dem ARGON Reservierungssystem und weiteren Diensten bereit, die externe Anfragen an das Reservierungssystem entgegennehmen und beantworten. Diese Struktur ermöglicht die parallele Verwendung verschiedener externer Schnittstellen. Derzeit werden externe Anfragen über einen Webservice entgegengenommen und über die IRes Schnittstelle an das ARGON Reservierungssystem weitergegeben. Die Verwendung von Adaptern ermöglicht dabei eine Abstraktion von der nach außen verwendeten Schnittstelle. Der in ARGON implementierte WebService zur Realisierung des externen Interfaces verwendet jedoch die gleiche Beschreibungssprache wie das interne Interface (IRes), sodass in diesem Falle kein Adapter erforderlich ist. Abbildung 1 zeigt die grundlegende Interfacestructur des ARGON Reservierungssystems.

Abbildung 1: Interfacestructur des Request Dispatchers. Die IRes Schnittstelle stellt lediglich ein internes Interface dar, welches verschiedene externe Interfaces bündelt.

Nach Entgegennahme einer externen Anfrage über die Webserviceschnittstelle werden durch den Dispatcher zunächst einige grundlegende Operationen durchgeführt. Als erster Schritt wird dazu die entgegengenommene Anfrage gegen das gültige XML Schema validiert. Schlägt die Validierung fehl handelt es sich um keine gültige Anfrage, was zu einer umge-
henden Ablehnung führt. Ist die Validierung erfolgreich, wird die Anfrage im nächsten Schritt in einer Datenbank persistiert. Nach Ausführung einer Policy-Überprüfung wird die Anfrage im nächsten Schritt an die entsprechende Kernkomponente zur weiteren Bearbeitung weitergegeben.

Der Request Dispatcher selbst arbeitet in mehreren parallelen Instanzen. Dies ermöglicht eine nebeneinäugige Bearbeitung mehrerer Anfragen gleichen oder unterschiedlichen Typs. Wie bereits skizziert können hierbei parallelisierbare und nicht-parallelisierbare Anfragen unterschieden werden. Zu letzterer Gruppe zählen alle Anfragen, welche Ressourcen im Netz benötigen, also einen Einfluss auf die Bearbeitung folgender nicht-parallelisierbarer Anfragen haben. Auch Anfragen dieses Typs werden zunächst parallel abgearbeitet. Ein Locking Mechanismus stellt in der weiteren Bearbeitung eine serielle Pfadberechnung sicher, was eine Mehrfachreservierung derselben Ressourcen verhindert. Da lediglich die Pfadberechnung seriell erfolgt, führt dies insgesamt zu einer verzahnten Bearbeitung der Anfragen.

Nach vollständiger Bearbeitung der Anfragen durch die Kernkomponenten wird die Anfrage erneut durch den Dispatcher persistiert. Hierbei werden Statusinformationen, die im System gesetzt wurden, gespeichert. Nach der erneuten Persistierung ist die Anfrage vollständig bearbeitet und wird mit Erstellung der Antwort abgeschlossen.

1.1.1.2 Reservation Handler

Aufgabe des Reservation Handlers ist die Koordination der Bearbeitung von Reservierungsanfragen. Je nach spezifizierten Parametern wird die Bearbeitung der Anfrage durch verschiedene Subkomponenten durchgeführt.

Im ersten Schritt wird die Anfrage an eine Validierungskomponente im System weitergegeben. Zu diesem Zeitpunkt ist bereits sichergestellt, dass die Anfrage gegenüber dem XML Schema valide, das heißt syntaktisch korrekt ist. Durch die hier durchgeführte Validierung wird sichergestellt, dass die zu bearbeitende Anfrage auch semantisch korrekt ist.

Bei dem folgenden Teil der Bearbeitung handelt es sich um Operationen, die für nicht-parallelisierbare Anfragen seriell ausgeführt werden müssen. Durch einen Locking-Mechanismus wird an dieser Stelle sichergestellt, dass die folgenden Operationen nicht parallel für mehrere Reservierungs-, Verfügbarkeits-, oder Änderungsanfragen ausgeführt werden.

Nach Setzen des Locks werden zunächst die in der Anfrage enthaltenen Dienste seriell betrachtet. Ziel ist die Bestimmung der maximal über die gestammte Servicedauer verfügbaren Ressourcen auf jedem Link des Netzwerks. Diese Topologie wird für die eigentliche Pfadberechnung benötigt und kann über die Resource Manager Komponente, beschrieben im Abschnitt 1.1.2.2, bestimmt werden. Hierzu wird die Start- und Endzeit des Services an die Resource Manager Komponente weitergegeben.

Die durch den Resource Manager gewonnene Topologie repräsentiert das Netz mit den maximal verfügbaren Ressourcen zwischen Start- und Endzeit des Dienstes. Da alle Verbindungen eines Dienstes zum gleichen Start- und Endzeitpunkt beginnen und enden sind diese genau dann umsetzbar, wenn eine Menge von parallel umsetzbaren Pfaden mit ausreichend Ressourcen für die gegebene Topologie existiert. Der Path Computer versucht eine solche Menge von Pfaden für die Verbindungen eines Services zu bestimmen. Hierzu wird die Topologie sowie der Dienst an die Path Computer Komponente weitergegeben. Sind alle Verbindungen des Dienstes im Netz umsetzbar, gilt die Bearbeitung des Dienstes als erfolgreich abgeschlossen. Die berechnete Route wird abgespeichert, damit sie zum Zeitpunkt der Schaltung des Dienstes zur Verfügung steht. Hierzu existiert intern für Reservierungsanfragen ein erweitertes Datenschema, welches die Verwaltung dieser Informationen ermöglicht. Die Zusammenführung dieses Datenschemas mit den durch den Dispatcher persistierten Anfragedaten er-
scheint hierbei sinnvoll, da dies inkonsistente Daten und die Verwaltung redundanter Daten reduziert. Darüber hinaus wird eine Zusammenführung der verteilten Daten vermieden.

Die Bestimmung der Topologie sowie die Berechnung der Routen werden für alle Dienste einer Reservierungsanfrage wiederholt. Kann für einen Dienst keine Route berechnet werden, ist die Bearbeitung der Reservierungsanfrage fehlgeschlagen. Andernfalls gilt die Routenberechnung für eine Reservierungsanfrage als erfolgreich.

Im Fall eines Dienstes des Typs „Malleable Reservation“ ist eine Modifikation der Anfragedebearbeitung notwendig. Hierbei werden nicht eine sondern alle sinnvollen zeitlichen Konfigurationen zu Beginn der Pfadberechnung bestimmt und während der eigentlichen Berechnung parallel betrachtet. Das grundlegende Vorgehen ist allerdings identisch. Informationen zu den verschiedenen Diensttypen finden sich zum Beispiel in [B2.4.1].

Im nächsten Schritt müssen die in der Routenberechnung zugewiesenen Ressourcen für alle Verbindungen aller Dienste einer Anfrage reserviert werden, sofern die Bearbeitung erfolgreich war. Hierzu werden die berechneten Routen sowie die Start- und Endzeitpunkte an den Resource Manager weitergegeben. Nach erfolgreicher Reservierung der Ressourcen kann das im Vorfeld gesetzte Lock wieder aufgehoben werden, was eine verzahnte Bearbeitung der Anfragen ermöglicht.

Die weitere Bearbeitung der Anfrage hängt nun von den gesetzten Parametern ab. Für alle Services, deren Bindparameter vollständig spezifiziert sind und deren AutoActivation Flag gesetzt ist, werden zwei Events im Scheduler eingeplant. Ersteres dient dazu, die Signalisierung für den Pfadaufbau zu planen, letzteres dient der Planung der Signalisierung des Pfadabbaus.

Die in der Anfrage enthaltenen Bindparameter werden in einer weiteren Datenstruktur hinterlegt, damit diese zur Signalisierung des Pfades zur Verfügung stehen. Durch Verwendung einer einheitlichen Datenstruktur, welche im Zuge der Umgestaltung des Reservierungssystems eingeführt werden soll, ist eine explizite Hinterlegung der Bindparameter nicht mehr notwendig.

Im letzten Schritt der Bearbeitung wird die Reservierungsanfrage an den Responder (vgl. Abschnitt 1.1.2.3) weitergeleitet. Diese erzeugt – abhängig vom Ergebnis der vorherigen Bearbeitungsschritte – eine Antwort auf die Reservierungsanfrage, die an den Anwender weitergeleitet wird. Hierzu wird die Antwort über die IRes Schnittstelle an die zugehörige externe Schnittstelle weitergeleitet.

1.1.1.3 Availability Handler

Der Availability Handler koordiniert die Bearbeitung von Verfügbarkeitsanfragen, die an das Reservierungssystem gestellt werden. Die Bearbeitung der Anfragen wird analog zur Bearbeitung von Reservierungsanfragen ausgeführt. Im Gegensatz zu Reservierungsanfragen werden allerdings keine Ressourcen im Netz reserviert, da Vorreservierungen durch ARGON nicht unterstützt werden. Statt einer verbindlichen Vorreservierung sollen die durch den Availability Request gewonnenen Daten [[B2.3.2] einen Hinweis auf die Durchführbarkeit einer möglichen späteren Reservierung geben. Dieser Mechanismus ist zur Koordination mehrere Reservierungen durch den Metascheduling-Dienst gedacht, bei dem ein gemeinsamer Startzeitpunkt für unterschiedliche Ressourcenarten (z.B. Rechen- und Netzwerkressourcen) gefunden werden soll, ohne hohen Arbeitsaufwand im Reservierungssystem zu erzeugen. Das heißt für die Implementierung in ARGON, dass ein erneuter Zugriff auf den Resource Manager nach erfolgreicher Routenberechnung nicht notwendig ist.

Im Allgemeinen lässt sich die Verfügbarkeitsanfrage parallelisieren, da diese keine Vorreservierung von angefragten Ressourcen ausführt. Damit spiegelt die Antwort auf eine Verfüg-
Skalierbarkeit des Netzwerkreservierungssystems ARGON

barkeitsanfrage nur die Verfügbarkeit der benötigten Ressourcen zum Anfragezeitpunkt wieder. Insbesondere besteht die Möglichkeit, dass Ressourcen einer gewünschten Reservierung zwischen Verfügbarkeitsanfrage und Reservierungsanfrage bereits reserviert wurden. Somit ist auch die parallele Bearbeitung der Anfrage unkritisch, da seitens des Reservierungssystems keine Garantie auf Verfügbarkeit der Ressourcen gegeben, also keine Vorreservierung ausgeführt wird. Dies erhöht die Skalierbarkeit des Systems bezüglich der Anzahl von Verfügbarkeitsanfragen im Vergleich zur ersten Version des ARGON Systems, da diese Art von Anfragen auf parallele Systeme verteilt werden können.

1.1.1.4 Modification Handler

Die Aufgabe des Modification Handlers ist die Koordination und Bearbeitung von Änderungsanfragen. Kernidee der Bearbeitung ist es, die im Netz reservierten Ressourcen vorläufig freizugeben und dann eine Neuberechnung der Route durchzuführen. Ist die Neuberechnung erfolgreich, werden die benötigten Ressourcen reserviert, andernfalls werden die für die ursprüngliche Reservierung benötigten Ressourcen wieder belegt.

Auch bei der Bearbeitung einer Änderungsanfrage wird zunächst eine semantische Validierung durchgeführt. Um zu gewährleisten, dass Ressourcen nicht doppelt vergeben werden, wird auch hier zunächst ein Lock für den Resource Manager gesetzt. Im nächsten Arbeitsschritt werden die durch die ursprüngliche Reservierung reservierten Ressourcen im Resource Manager freigegeben. Diese Operation sollte hierbei in einer temporären Kopie erfolgen, um Änderungen im Falle der Nichtumsetzbarkeit der Änderungsanfrage leicht rückgängig machen zu können. Die im Resource Manager gespeicherten Informationen bleiben davon zunächst unberührt. Derzeit werden Änderungen direkt auf der Produktiv-Datenstruktur ausgeführt.

Nach Freigabe der Ressourcen wird die Änderungsanfrage ähnlich zu einer Reservierungsanfrage bearbeitet. Dienste und Verbindungen, die als gelöscht markiert sind, werden in dieser Bearbeitungsphase ignoriert. Unveränderte und modifizierte Dienste und deren Verbindungen werden zusammen durch das Reservierungssystem neu eingeplant.

Können alle nicht gelöschten Dienste und Verbindungen durch den Path Computer eingeplant werden, ist die Änderungsanfrage erfolgreich ausgeführt worden. In diesem Fall wird die neue Ressourcenbelegung im Resource Manager gespeichert und der Lock aufgehoben. Im Folgenden werden die in der ursprünglichen Reservierungsanfrage gespeicherten Daten aktualisiert. Dies betrifft sowohl geänderte Start- und Endzeiten bzw. Quell und Zielknoten, wie auch geänderte interne Informationen über die Route im Netz.

Ist mindestens eine Verbindung nicht umsetzbar, ist die Änderungsanfrage fehlgeschlagen. Durch eine Speicherung der Ressourcenauslastungsinformationen in einer temporären Datenstruktur können diese durch den Resource Manager einfach verworfen werden. Auch hier kann der gesetzte Lock aufgehoben werden. Soll eine Reservierung bei Nichtumsetzbarkeit der Änderung gelöscht werden, kann nun die gleiche Prozedur wie zur Ausführung einer Löschanfrage durchgeführt werden.

Im letzten Schritt der Bearbeitung wird eine Antwort auf die Änderungsanfrage generiert und versendet. Hierzu wird der Request sowie weitere benötigte Informationen an die Responderkomponente weitergegeben.

1.1.1.5 Bind Handler

Der Bind Handler hat die Aufgabe Bindinformationen, die durch eine entsprechende Anfrage übermittelt werden, in den Request einzutragen. Zu Beginn der Anfragebearbeitung wird auch hier zunächst eine semantische Validierung durchgeführt. Ist diese erfolgreich, werden die übermittelten Parameter in die Anfrage eingetragen.
Werden Bindparameter für eine Reservierung übertragen, deren AutoActivation Flag gesetzt ist, muss die Signalisierung der Pfadauf- und -abbau automatisch durch ARGON ausgelöst werden. Waren vorher bereits Bindparameter spezifiziert, ist eine Einplanung bereits vorher erfolgt.

Auch hier wird die Bearbeitung der Anfrage durch Generierung einer geeigneten Antwort abgeschlossen. Wird eine korrekte Bindanfrage zu einer angenommenen Reservierung spezifiziert, kann bei der Bearbeitung dieser Anfrage kein Fehler auftreten.

1.1.1.6 Query Handler
Die Bearbeitung von Abfrageanfragen wird durch den Query Handler koordiniert. Die Bearbeitung einer solchen Anfrage beginnt auch hier zunächst mit der semantischen Validierung. Ist diese erfolgreich, wird die zugehörige Reservierungsanfrage durch das System ausgelesen und die darin enthaltenen Reservierungsdaten bestimmt. Diese werden an die Responderkomponente weitergegeben, um eine gültige Antwort auf die Anfrage zu generieren.

1.1.1.7 Cancel Handler
Aufgabe des Cancel Handlers ist die Bearbeitung von Löschanfragen. Im ersten Schritt der Bearbeitung erfolgt die semantische Überprüfung der Löschanfrage. Ist diese Überprüfung erfolgreich, werden die durch die zugehörige Reservierung belegten Ressourcen mit Hilfe der Resource Manager Komponente freigegeben. Die Verwendung eines Locks ist an dieser Stelle nicht notwendig, da sich durch Ausführung einer Löschanfrage die verfügbaren Ressourcen im Allgemeinen vergrößern. Zu keinem Zeitpunkt werden sich diese verkleinern. Damit ist es nicht möglich, dass durch die parallele Ausführung einer Löschanfrage Ressourcen mehrfach zugeordnet werden.

Auch hier wird die Bearbeitung der Anfrage durch Generierung einer Antwort durch die Responderkomponente abgeschlossen.

1.1.1.8 Activation Handler
Aktivierungsanfragen ermöglichen das explizite Signalisieren eines Pfadaufbaus. Dies ist immer dann notwendig, wenn das AutoActivation Flag innerhalb der zugehörigen Reservierungsanfrage nicht gesetzt worden ist.

Auch der Activation Handler validiert die eingehende Anfrage zunächst mit Hilfe der Validierungskomponente semantisch. Ist die Validierung erfolgreich werden Pfadaufbau und Pfadausbau in der Schedulerkomponente eingeplant. Im letzten Schritt der Bearbeitung wird die Anfrage zur Generierung und Versendung einer Antwort an die Responderkomponente weitergegeben. Damit ist ein Activation Request vollständig bearbeitet.

1.1.2 Subkomponenten
Die als Subkomponenten bezeichneten Teile des Systems führen grundlegende Operationen aus, die zur Bearbeitung von verschiedenen Anfragetypen benötigt werden. Diese Komponenten und ihre Funktionsweise sollen in den folgenden Abschnitten skizziert werden.

1.1.2.1 Validator
Aufgabe der Validatorkomponente ist es, eingehende Anfragen des Systems auf ihre semantische Korrektheit zu überprüfen. Eine syntaktische Überprüfung gegen das gültige XML-Schema wird durch den Validator nicht ausgeführt. Diese Überprüfung wird bereits bei Entgegennahme der Anfrage durch den Dispatcher ausgeführt, um eine fehlerfreie Persistierung der Anfrage zu ermöglichen.
In der semantischen Überprüfung einer Anfrage müssen alle logischen Bedingungen, die nicht mit Hilfe des Schemas ausgedrückt werden können, überprüft werden. Diese Überprüfung ist äußerst wichtig, da keine weitere Überprüfung in den einzelnen Subkomponenten ausgeführt wird.

Ein einfaches Beispiel für eine nicht im Schema spezifizierbare Bedingung ist die logische Abhängigkeit von Start- und Endzeitpunkt im Falle von Reservierungs-, Verfügbarkeits- und Änderungsanfragen. Offensichtlich sind solche Anfragen nur dann sinnvoll, wenn der spezifizierte Startzeitpunkt vor dem spezifizierten Endzeitpunkt liegt. Eine solche logische Abhängigkeit kann nicht über das XML-Schema definiert werden. Aus diesem Grund müssen diese und ähnliche Bedingungen mit Hilfe der Validatorkomponente überprüft werden.

Ähnliche Bedingungen wurden auch für alle andere Typen von Anfragen definieren, die ebenfalls durch den Validator bearbeitet werden.

1.1.2.2 Resource Manager

Aufgabe des Resource Managers ist es, Informationen über die Auslastung von Ressourcen innerhalb des aktuellen Book-ahead Intervalls zu verwalten, bereitzustellen und zu modifizieren. Hierzu werden die belegten Ressourcen für verschiedene Zeitabschnitte variabler Länge, welche das Book-ahead Intervall vollständig abdecken, verwaltet. Die Länge eines solchen Zeitabschnittes ist als ganzzahliges Vielfaches einer Mindestzeitlottlänge gegeben. Neben diesen Informationen werden zusätzlich Informationen über die Netzttopologie verwaltet.

Zur Bereitstellung von Ressourcenauslastungsinformationen fragt der Resource Manager die benötigten Informationen zunächst aus einer Datenbank ab. Während im Reservierungssystem ARGON keine Pufferung der Daten stattfindet, empfiehlt es sich im Rahmen des Redesigns, diese zu einer Liste von Zeitslots variabler Länge aufzubereiten und intern zu verwalten. Dieses Vorgehen dient dazu, temporäre Änderungen an der Ressourcenauslastung verwalten zu können, ohne diese dauerhaft in der Datenbank persistieren zu müssen. Da in der eigentlichen Routenberechnung keine zeitlichen Aspekte betrachtet werden, stellt der Resource Manager für ein gegebenes Zeitintervall eine Topologie mit den maximal über die gesamte Anfragezeit verfügbaren Ressourcen im Netz bereit.

Durch die Bearbeitung von Reservierungs-, Lösch- und Änderungsanfragen ändert sich im Allgemeinen die Auslastung der Ressourcen im Netz. Der Resource Manager ermöglicht es, angenommene Reservierungen in die zukünftige Ressourcenauslastung einzuplanen. Ist die Bearbeitung einer Anfrage abgeschlossen und sollen die Informationen dauerhaft gespeichert werden, können diese Informationen in der Datenbank persistiert werden. Auch hierzu stellt der Resource Manager die benötigten Funktionen zur Verfügung. Im Falle einer fehlgeschlagenen Anfrage sind keine weiteren Änderungen notwendig, da diese erst nach erfolgreichem Abschluss in der Datenbank persistiert werden und die Informationen vor jeder Bearbeitung durch den Resource Manager aus der Datenbank neu eingelesen werden.

1.1.2.3 Responder

Die Responder Komponente erzeugt Antworten auf Anfragen, die an das System gestellt wurden. Der genaue Aufbau der Antworten hängt dabei vom Typ der Anfrage und dem Ergebnis der Anfragebearbeitung ab. Der genaue Aufbau einer Antwort kann in der Interfacespezifikation Program Documentation of the Reservation System Interface, [B2.3.2, nachgelesen werden.

Als Antwort wird ein gegen das Schema valider XML String erzeugt, der die Anfrage gemäß der Interfacespezifikation beantwortet. Dieser wird dann über das IRes Interface an das externe Interface übermittelt, an das die Anfrage gestellt wurde.
1.1.2.4 Scheduler
Aufgabe der Schedulerkomponente ist es, die geplanten Reservierungen zum richtigen Zeitpunkt im Netz umzusetzen. Dies ist notwendig, da die im VIOLA-Netz eingesetzten Komponenten kein Verständnis von Zeit besitzen, der Pfadauf- bzw. -abbau also von extern angestoßen und zu den richtigen Zeitpunkten durchgeführt werden muss.

Bei erfolgreicher Routenberechnung werden zwei Ereignisse, der Routenauf- und -abbau in der Schedulerkomponente eingeplant. Dadurch wird die Signalisierung zeitlich vom der weiteren Anfragebearbeitung getrennt.

1.2 Zusammenwirken der Einzelkomponenten
Die Verwendung eines JMS Topics zur Implementierung des externen Interfaces ermöglicht es, mehrere parallele Instanzen des Request Dispatchers einzusetzen, sofern Anfragen parallel bearbeitet werden sollen. Im ARGON Reservierungssystem wird hier lediglich eine einzelne Instanz eingesetzt, was zu einer seriellen Bearbeitung von Reservierungsanfragen führt. Durch die Verwendung mehrerer paralleler Instanzen könnte durch das Redesign die in den vorherigen Abschnitten beschriebene Parallelisierung erzielt werden. Der folgende Abschnitt beschreibt die Bearbeitung von Anfragen, wie sie nach einem Redesign sinnvoll ausgeführt werden könnte.

1.2.1 Parallelisierung der Anfragebearbeitung
Die Bearbeitung der Anfragen erfolgt zunächst parallel, wobei dies unabhängig davon ist, ob eine parallelisierbare oder eine nicht-parallelisierbare Anfrage bearbeitet wird. Das unterschiedliche Vorgehen bei der Bearbeitung der Anfragen tritt erst im Laufe der Bearbeitung auf. Parallelisierbare Anfragen werden durch das ARGON Reservierungssystem vollständig parallel abgearbeitet. Diese werden insbesondere auch während der Bearbeitung von nicht-parallelisierbaren Anfragen ausgeführt.

Die Bearbeitung von nicht-parallelisierbaren Anfragen erfolgt zunächst auch parallel. Dies ermöglicht das parallele Ausführen vorbereitender Arbeitsschritte wie die Validierung von Anfragen. Operationen nicht-parallelisierbarer Anfragen, die zu Änderungen der Ressourcenauslastung führen, werden durch das Reservierungssystem seriell ausgeführt, sofern neue Ressourcen reserviert werden. Dies ist notwendig, um eine doppelte Reservierung selbiger Ressourcen zu verhindern. Dazu wird die Resource Manager Komponente zunächst gesperrt, wenn diese nicht für eine andere, nicht-parallelisierbare Anfrage benötigt wird. Darauf folgende, nicht-parallelisierbare Anfragen unterbrechen bei gesetztem Lock ihre Ausführung, bis dieser wieder zur Verfügung steht. Die Anfrage, welche zur Setzung des Locks geführt hat, kann so mit einem exklusiven Zugriff auf die benötigten Subkomponenten bearbeitet werden. Ist die Bearbeitung der Anfrage erfolgreich, kann der Lock aufgehoben werden. Operationen, die nach dem Entfernen des Locks ausgeführt werden, können dann wieder parallel zu beliebigen Anfragen ausgeführt werden. Abbildung 2 skizziert die Bearbeitung nicht-parallelisierbarer Anfragen schematisch.
Abbildung 2: Die Verwendung eines Locks verhindert die parallele Ausführung nicht-parallelisierbarer Operationen, auch wenn die zugehörigen Anfragen unabhängig voneinander bearbeitet werden. Insgesamt führt dieses Verfahren zu einer verzahnten Bearbeitung nicht-parallelisierbarer Anfragen. Operationen, welche parallel ausgeführt werden können, werden ohne Blockierung auf diese Weise abgearbeitet, lediglich Operationen die einen exklusiven Zugriff auf Subkomponenten benötigen werden seriell ausgeführt und vor dem Zugriff anderer Bearbeitungsinstanzen geschützt. Insgesamt könnte das Redesign im Vergleich zum eingesetzten Reservierungssystem so zu einem deutlich gesteigerten Grad der Parallelisierung beitragen.
2 ARGON Eingabeparameter

In Bezug auf die Skalierbarkeit des Reservierungssystems können im Wesentlichen zwei verschiedene Aspekte betrachtet werden. Diese sind durch die Laufzeit, die zur Bearbeitung einer Reservierungsanfrage benötigt wird und der Fehlerresistenz des Gesamtsystems gegeben.

Die Auswirkung von und die Absicherung vor Fehlern soll in diesem Dokument nur kurz in Abschnitt 2.1 betrachtet werden. Stattdessen liegt der Schwerpunkt der Betrachtungen auf der Laufzeit, die zur Bearbeitung von Anfragen durch das Reservierungssystem benötigt wird. Hierbei ist insbesondere die Zeit, die zur Bearbeitung von Reservierungsanfragen benötigt wird, von Interesse. Diese hängt im Allgemeinen von den verschiedenen Eingabeparametern ab, die als Grundlage für die Routenberechnung dienen.

Im Wesentlichen können drei verschiedene Aspekte betrachtet werden, die einen direkten Einfluss auf die Laufzeit der Reservierungsanfragenbearbeitung haben. Der erste wichtige Aspekt ist durch die Größe und den Aufbau der Topologie, die durch das Reservierungssystem verwaltet werden soll, gegeben. Es ist offensichtlich, dass eine Vergrößerung der Topologie die Ausführungszeit einer Pfadberechnung im Allgemeinen erhöht. Es ist zu beachten, dass eine Vergrößerung der Topologie die Ausführungszeit einer Pfadberechnung im Allgemeinen erhöht. Eine genauere Betrachtung der Abhängigkeiten soll in Abschnitt 2.3 durchgeführt werden.

Der zweite wichtige Aspekt ist durch die Anzahl der Reservierungen, die durch das Reservierungssystem verwaltet werden, gegeben. Insbesondere für die Laufzeit, welche zur Bestimmung der verfügbaren Ressourcen benötigt wird, spielt dieser Wert eine wichtige Rolle. Sie wird im Allgemeinen mit steigender Anzahl der zu verwaltenden Reservierungen zunehmen. Abschnitt 2.3 skizziert verschiedene Verfahren und deren Auswirkungen, um die im Book-ahead Interval reservierten Ressourcen zu verwalten. Auch die Parameter, welche in der Anfrage spezifiziert werden, haben einen Einfluss auf die Bearbeitungszeit von einzelnen Anfragen und somit auf die Skalierbarkeit des Reservierungssystems.

Im Falle von Malleable Reservations ist ein einfaches Beispiel durch das Zeitfenster gegeben, in welchem die Reservierung eingeplant werden kann. Vergrößert sich dieses Zeitfenster bei gleichbleibenden Randbedingungen, vergrößert sich auch die Anzahl der Konfigurationen, die während des Preprocessings und der Routenberechnung betrachtet werden müssen. Eine genaue Betrachtung dieses Aspekts erfolgt in Abschnitt 2.5.

2.1 Fehlertoleranz

Das ARGON Reservierungssystem ist als zentrales Ressourcen-Reservierungssystem konzipiert. Routen im Netz werden dabei nicht verteilt, sondern durch eine zentrale Komponente im Netz berechnet. Obwohl dieses Verfahren mit einer Reihe von Vorteilen verbunden ist, gibt es auch Nachteile, die insbesondere die Skalierbarkeit des Gesamtsystems betreffen.

2.1.1 Ausfall einzelner Netzkomponenten

Eine wichtige Gruppe von Fehlern ist der Ausfall einzelner Links oder Router im Netzwerk. Die Wahrscheinlichkeit für solche Fehler wird im Allgemeinen mit der Größe der verwendeten Topologie wachsen. Treten solche Fehler auf, ist es zunächst wichtig, dass das Reservierungssystem über diese informiert wird. Dies kann zum Beispiel sichergestellt werden, indem Topologieinformationen mittels OSPF aus dem Netz gewonnen werden. Ist das Reservierungssystem über den Ausfall der betroffenen Links informiert, treten verschiedene Herausforderungen auf.

Der erste wichtige Punkt ist es, dass die betroffenen Links im Netz nicht für die Planung weiterer Reservierungen verwendet werden, die die Ausfallzeit des Links schneiden. Hierbei stellt sich die Herausforderung, wie die Ausfallzeit eines Links oder eines Routers bestimmt...
werden kann. Im Allgemeinen wird diese je nach Ursache stark variieren. Handelt es sich beispielsweise um ein Softwareproblem oder eine Fehlkonfiguration, so kann dieses in einigen Fällen vielleicht in wenigen Minuten behoben werden. Handelt es sich dagegen beispielsweise um einen Leitungsaufall, der durch einen physischen Schaden an der Leitung verursacht wird, so kann das Beheben des Schadens je nach Lage der Leitung mehrere Wochen in Anspruch nehmen. In der derzeitigen Implementierung des Reservierungssystems können Ausfälle von Links sowie die Dauer des Ausfalls in der Topologie manuell in der Topologie konfiguriert werden. Die automatische Bestimmung der Ausfallzeit stellt hierbei einen Punkt dar, der in weiteren Untersuchungen genauer betrachtet werden kann. Diese Funktion kann auch verwendet werden, um Links oder Komponenten für geplante Wartungsarbeiten von Reservierungen auszuschließen. Dadurch ist das Herunterfahren während der Wartung von Netzkomponenten im Allgemeinen nicht mehr notwendig.

2.1.2 Fehler im Reservierungssystem

Neben Fehlern im Netz die zu fehlerhaften Pfadberechnungen durch das System führen können, können auch Fehler im Reservierungssystem selbst auftreten. Designbedingt stellt das ARGON Reservierungssystem einen Single Point of Failure dar, da ein Ausfall des Reservierungssystems Reservierungen im gesamten System betreffen und je nach Schwere des Fehlers verhindern. Verschiedene Möglichkeiten, um hier eine Absicherung gegen Fehler im eigentlichen Reservierungssystem zu erreichen sollen in Abschnitt 3 dieses Dokumentes erörtert werden.

2.2 Laufzeitannteile zur Bearbeitung von Reservierungsanfragen

Die Bearbeitung einer Reservierungsanfrage kann insgesamt in drei Phasen unterteilt werden. In der ersten Phase, der Preprocessing Phase, wird eine Topologie erstellt, welche die maximalen, im gesamten Zeitintervall verfügbaren Ressourcen repräsentiert. Im Falle einer Advance Reservation muss zur Routenberechnung lediglich eine einzelne Topologie bestimmt werden.

Dieses Vorgehen ist für Malleable Reservations nicht ausreichend. Für Malleable Reservations gibt es im Allgemeinen zwei weitere Freiheitsgrade, die vom System für die Bestimmung der Lösung berücksichtigt werden müssen. Der erste Freiheitsgrad ist durch den Startzeitpunkt gegeben, an dem die Reservierung beginnen soll. Der zweite ist durch die verwendete Bandbreite bzw. durch die Übertragungsdauer gegeben. Die Betrachtung dieser Freiheitsgrade führt dazu, dass eine Menge von Topologien bestimmt werden muss, welche alle zur Bestimmung der Best-fit Lösung innerhalb der Routenberechnung zu betrachten sind.

In der zweiten Phase, der Path Computation, wird die eigentliche Pfadberechnung ausgeführt. Hierzu wird eine gültige Konfiguration bestimmt, welche aus einem Start- und einem Endzeitpunkt sowie einem Pfad im Netz zusammengesetzt ist. Als Grundlage für die Berechnung dienen hier die im Preprocessing bestimmten Topologien, so dass eine Betrachtung zeitlicher Aspekte hier nicht notwendig ist.

In der dritten Phase der Routenberechnung, welche als Postprocessing bezeichnet wird, werden die Ressourcen im Netz für die gegebenen oder berechneten Zeitintervalle reserviert. In den folgenden Abschnitten wird die Auswirkung der Eingabeparameter auf die verschiedenen Laufzeitannteile betrachtet.

2.3 Netztopologien

Die Topologie in der Reservierungen durch das ARGON Reservierungssystem umgesetzt werden, wird durch einen Graph charakterisiert. Im Folgenden soll davon ausgegangen werden, dass es sich hierbei um einen einfachen Graphen \( G \) handelt, der aus einer Menge von
Knoten V und einer Menge von Kanten E zusammengesetzt ist. Die Anzahl der Kanten m ist hierbei durch $O(n^2)$ beschränkt, wobei n die Anzahl der Knoten repräsentiert. Die folgenden Abschnitte diskutieren das Laufzeitverhalten für eine steigende Anzahl Knoten und Kanten im Graph.

2.3.1 Preprocessing

Die Auswirkung der Größe der Netztopologie auf die Zeit, die für das Preprocessing benötigt wird, hängt davon ab, wie die über die Zeit im Netz belegten Ressourcen verwaltet werden. Eine genauere Betrachtung der verschiedenen Verfahren findet sich in Abschnitt 2.4.1.

Werden lediglich die im Netz berechneten Routen gespeichert und diese Informationen verwendet, um Informationen über die Netzauslastung zu gewinnen, so liegt keine direkte Abhängigkeit zwischen der Topologiegröße und der Zeit die für das Preprocessing benötigt wird, vor. Stattdessen ergibt sich lediglich eine Abhängigkeit von der durchschnittlichen Pfadlänge der berechneten Pfade und der Anzahl der angenommenen Reservierungen, da jeder Link eines Pfades der Verbindungen betrachtet werden muss, die das Anfrageintervall überschneiden. In diesem Fall ist die Vergrößerung der Netztopologie für die Preprocessingphase laufzeitunkritisch.

Ein anderer Ansatz ist es, die belegten Ressourcen der Links im Netz über die Zeit kumuliert vorzuhalten. Damit ist die Größe der Datenstruktur, die zur Bestimmung der Ressourcenauslastung gepflegt werden muss, linear von der Anzahl der Links im Netz abhängig. Sofern es sich um ein einfaches Netzwerk handelt, ist die Anzahl der Links wiederum quadratisch durch die Anzahl der Knoten im Netz beschränkt. Der Worst Case tritt hierbei in einem vollvermaschten Netz auf.

Aus Gründen, die in Abschnitt 2.4 näher erläutert werden, wird dieses Verfahren auch im ARGON Reservierungssystem verwendet. Insgesamt muss hierbei jeder Link zur Bestimmung einer Konfiguration genau einmal betrachtet werden. Hierdurch ergibt sich für die Laufzeit des Preprocessings eine lineare Abhängigkeit zur Anzahl der Links im Netz, was aus Sicht der Skalierbarkeit des Gesamtsystems zu vertreten ist.

2.3.2 Path Computation

Die Größe der zu betrachtenden Netztopologie ist einer der Kernfaktoren, die die Laufzeit der eigentlichen Pfadberechnung bestimmen. Die genaue Abhängigkeit ist hierbei durch die Randbedingungen und der verwendeten Kostenmetrik vorgegeben, die im Rahmen der Pfadberechnung berücksichtigt werden sollen.

Im Allgemeinen kann davon ausgegangen werden, dass eine additive Distanzmetrik zur Bestimmung der Kosten von Pfaden verwendet wird. Betrachtet der Pfadalgorithmus zur Routenberechnung lediglich Randbedingungen welche Bottleneck Metriken unterliegen, kann zur Pfadbestimmung ein modifizierter Dijkstra Algorithmus verwendet werden. Dieser ist insgesamt durch eine Laufzeit von $O(n \log n + m)$ beschränkt, wobei n der Anzahl der Knoten und m der Anzahl der Kanten im Graphen entspricht, sofern der Algorithmus mittels Fibonacci-Heap implementiert wird. Diese Laufzeitschranke ist für das ARGON Reservierungssystem in der aktuellen Implementierung gültig, da diese lediglich die Bandbreite als spezifizierbare Randbedingungen betrachtet, welche einer Bottleneck Metrik unterliegt.

Die Erweiterungen des Reservierungssystems, die im Rahmen des Redesigns angedacht sind, beachten neben der durch eine Verbindung benötigte Bandbreite auch eine obere Schranke für die zulässige maximale Ende-Zu-Ende Verzögerung eines Pfades. Hierbei handelt es sich um
eine Randbedingung, die einer additiven Metrik unterliegt. Dies führt dazu, dass es in der Pfadberechnung nicht mehr ausreichend ist, den bezüglich der Kostenmetrik kürzesten Pfad, welcher ausreichend Bandbreite zur Verfügung stellt, zu betrachten.

Abbildung 3: Beispiel für eine durch einen Dijkstra Algorithmus berechnete Route, die den Anforderungen einer additiven Randbedingung – in diesem Fall der Ende-Zu-Ende Verzögerung – nicht entspricht. Hierdurch würde keine Lösung gefunden werden, obwohl ein Pfad, der die Randbedingungen erfüllt, existiert.

Abbildung 3 zeigt ein Beispiel für eine Topologie, in der die angefragte Verbindung bei Betrachtung des kürzesten Pfades nicht ermittelt werden kann. Wird wie in diesem Beispiel der Wert Eins als Kosten für jeden Link verwendet, werden keine Pfade länger als zwei betrachtet, da diese höhere Kosten als der kürzeste Pfad verursachen. Damit kann ein solcher Algorithmus im obigen Beispiel keinen Pfad finden, der zwischen Quelle und Ziel eine Ende-Zu-Ende Verbindung von weniger als 6ms bereitstellt, obwohl ein solcher Pfad existiert. Vielmehr erfordert die Berücksichtigung der additiven Kostenmetrik und der additiven Ende-Zu-Ende Verzögerung die Betrachtung aller Pfade im Netz, die am gegebenen Quell- und Zielpunkt starten und enden.

Geht man von einem einfachen Graphen als Modell der Netzwerktopologie aus, so ist die Anzahl der verschiedenen Pfade im Netz durch

$$O(k \max \{d_G^*(v) : v \in V\} + k \log k) \quad |k = \sum_{i=2}^{n-2} \frac{(n-i)!}{(n-2-i)!}$$

beschränkt. Auch hier beschreibt n die Anzahl der Knoten im Graph, $d_G^*(v)$ bezeichnet den maximalen Ausgangsgrad des Knotens v aus dem Graphen G. Der Worst Case tritt hierbei in einem vollvermaschten Netz auf.

Auch wenn die Anzahl der zu betrachtenden Pfade im Netz im Allgemeinen deutlich geringer als die obere Schranke sein wird, stellt das im Worst Case exponentielle Wachstum der zu betrachtenden Pfade ein Problem für die Skalierbarkeit des Reservierungssystems dar. Dies ist dadurch bedingt, dass die vollständige Route zentral berechnet wird. Mögliche Ansätze zur Vermeidung dieses Problems werden in Abschnitt 3 diskutiert.

**2.3.3 Postprocessing**

Für die Laufzeit der Postprocessing Phase ergeben sich ähnliche Abhängigkeiten wie für die Preprocessing Phase. Hierbei muss allerdings nicht zwischen Advance und Malleable Reservation unterschieden werden, da lediglich Ressourcen auf einem Pfad für eine Konfiguration reserviert werden. Dies ist unabhängig von der Anzahl der während der Routenberechnung betrachteten Konfigurationen.

Wird die Ressourcenauslastung der einzelnen Links nicht kumuliert vorgehalten, führt dies zur Unabhängigkeit von der jeweiligen Netztopologie. Wird die Ressourcenauslastung der einzelnen Links in einer weiteren Datenstruktur kumuliert verwaltet, ergibt sich eine lineare
Abhängigkeit zur Anzahl der Links im Netz. Ähnlich wie bei der für das Preprocessing benötigten Zeit ist dieses Laufzeitverhalten aus Sicht der Skalierbarkeit des Gesamtsystems zu vertreten.

### 2.4 Anzahl angenommener Reservierungsanfragen

Ein weiterer wichtiger Parameter, der die Laufzeit zur Bearbeitung einer Anfrage bestimmt, ist die Anzahl der angenommenen Reservierungsanfragen. Wie im folgenden Abschnitt beschrieben kann dieser Parameter je nach Art der Implementierung einen wesentlichen Einfluss auf die benötigte Preprocessingzeit haben.

#### 2.4.1 Preprocessing

Grundsätzlich gibt es verschiedene Möglichkeiten die reservierten Ressourcen im Netz zu verwalten. Das Verfahren, welches zur Verwaltung der Informationen verwendet wird, hat dabei direkten Einfluss auf die Zeit, die für das Preprocessing benötigt wird. Diese Zeit macht insbesondere für Malleable Reservations den größten Teil der benötigten Laufzeit aus. Aus diesem Grund sollen im Folgenden drei verschiedene Ansätze, deren Auswirkung, sowie deren Vor- und Nachteile skizziert werden.

##### 2.4.1.1 Verwaltung mittels Konfigurationen

Für die Signalisierung des Pfadauf- und -abbaus ist es notwendig, Informationen über den Startzeitpunkt der Reservierung, deren Endzeitpunkt, sowie die reservierten Ressourcen und den berechneten Pfad im Netz vorzuhalten. Diese Informationen können aber auch genutzt werden, um die maximal verfügbaren Ressourcen der einzelnen Links über den gesamten Zeitraum zwischen Start- und Endzeitpunkt einer Reservierungsanfrage zu bestimmen. Ein erheblicher Vorteil des Verfahrens ist es, dass keine weitere Datenstruktur und somit auch kein weiterer Speicher benötigt wird.

Für die folgenden Betrachtungen wird angenommen, dass die angenommenen Reservierungen $R$ sowie die zu bearbeitende Reservierungsanfrage $r_{\text{new}}$ semantisch korrekt spezifiziert sind. Insbesondere heißt das, dass die Startzeit vor der Endzeit liegt, also

\[
\text{res}_{\text{new}}^{\text{start}} < \text{res}_{\text{new}}^{\text{end}} \land \text{res}_{\text{old}}^{\text{start}} < \text{res}_{\text{old}}^{\text{end}} \quad \forall \text{res} \in R.
\]

Die Bestimmung der maximal über den gesamten Zeitraum verfügbaren Ressourcen kann in drei Schritten durchgeführt werden. Im ersten Arbeitsschritt werden zunächst alle durch das System angenommenen Reservierungen betrachtet. Ziel ist es die Teilmengen von Reservierungen zu bestimmen, die das Zeitintervall überschneiden, in der die neue Reservierungsanfrage eingeplant werden soll. Bezeichnet $\text{res}_{\text{old}}$ eine bereits angenommene Reservierung, $\text{res}_{\text{new}}$ eine eingehende Reservierungsanfrage und $\text{res}_{\text{start}}$ bzw. $\text{res}_{\text{end}}$ den Start- und Endzeitpunkt der Reservierung res, so liegt eine Überschneidung genau dann vor, wenn

\[
\text{res}_{\text{old}}^{\text{start}} < \text{res}_{\text{new}}^{\text{end}} \land \text{res}_{\text{new}}^{\text{start}} < \text{res}_{\text{old}}^{\text{end}}
\]
Skalierbarkeit des Netzwerkreservierungssystems ARGON

gilt. Hieraus kann die Teilmenge $R'$ bestimmt werden, die alle Reservierungen enthält, die die neue Anfrage $r_{\text{new}}$ überschneiden. Abbildung 4 zeigt ein Beispiel für überschneidende und nicht überschneidende Reservierungen und deren zeitliche Lage in Bezug auf eine Reservierungsanfrage $r_{\text{new}}$.

Abbildung 4: Angenommene Reservierungen, die durch eine zu bearbeitende Reservierungsanfrage überschnitten wird.

Im zweiten Arbeitsschritt müssen die für jeden Link maximal belegten Ressourcen bestimmt werden. Hierzu ist es ausreichend, an den Startzeitpunkten der Reservierungen $R'$ die kumulierten, belegten Ressourcen zu betrachten. Voraussetzung für diese Art von Betrachtung ist die Reservierung einer konstanten Bandbreite über die gesamte Dauer der Reservierung. Diese Bedingung wird bereits durch die Spezifikation des ARGON.UI User Interfaces, vergleiche [[B2.4.1]], sichergestellt. Die Betrachtung der Startzeitpunkte ist dabei ausreichend, da dies die einzigen Zeitpunkte sind, an denen sich die reservierten Ressourcen erhöhen können. Die im Intervall maximal belegten Ressourcen sind dann durch das Maximum der kumulierten, belegten Ressourcen an den Startzeitpunkten gegeben. Der maximal belegte Wert in einem Zeitintervall für Ressourcen, die nicht einer Bottleneck-Metrik unterliegen, kann nicht mittels Maximumbildung bestimmt werden. Dieser Wert muss abhängig von der jeweiligen Metrik bestimmt werden. Im Fall der End-zu-End Verzögerung bietet sich die Betrachtung der maximalen Linkverzögerungen an, welche als konstant betrachtet werden kann. Wie in [[MS01]] beschrieben hängt die Verzögerung eines Links im Allgemeinen von einer Reihe von Parametern ab, deren genaue Werte nicht vorhergesagt werden kann. Trotzdem ist es möglich für diese eine obere bzw. untere Schranke zu bestimmen, womit auch die obere Schranke für die Verzögerung spezifiziert werden kann. Diese Art von Ressource wird nicht auf den jeweiligen Links reserviert, stattdessen wird lediglich überprüft, ob die kumulierten Linkverzögerungen nicht überschnitten werden.

In der dritten Phase werden die im angefragten Intervall maximal verfügbaren Ressourcen bestimmt, die über das gesamte Anfrageintervall zur Verfügung stehen. Hierzu werden die bereits bestimmten Informationen über die im angefragten Intervall maximal belegten Ressourcen und Informationen über die auf den Links maximal verfügbaren Ressourcen benötigt. Letztere sind im Allgemeinen mit den Topologieinformationen gegeben. Aus diesen Informationen können die über das gesamte Anfrageintervall maximal verfügbaren Ressourcen leicht bestimmt werden.

Eine genauere Betrachtung des Verfahrens zeigt, dass sich die für das skizzierte Verfahren benötigte Laufzeit quadratisch zur Anzahl der angenommenen Reservierungen verhält. Insbesondere für Malleable Reservations, in denen eine Vielzahl von Konfigurationen betrachtet werden muss, erscheint dieses Laufzeitverhalten nicht akzeptabel. Hierbei kann es bei einer hohen Anzahl von eingeplanten Reservierungen zu Skalierbarkeitsproblemen kommen.

Ein weiteres Problem ergibt sich durch die Anzahl der Reservierungen, die durch das Reservierungssystem verwaltet werden muss. Im Regelfall sind innerhalb einer Domain lediglich die verfügbaren Ressourcen, nicht aber die Anzahl der annehmbaren Reservierungen beschränkt. Dies bedeutet gleichzeitig, dass die Zeit, welche für das Preprocessing benötigt
wird, nicht nach oben beschränkt ist. Damit ist es ebenfalls nicht möglich, eine obere Schranke für die Bearbeitungszeit einer Reservierungsanfrage anzugeben. Eine solche Schranke ist im Allgemeinen allerdings erwünscht. Möglichkeiten zur Beschränkung bestünden lediglich darin, die minimal pro Reservierung reservierbaren Ressourcen und eine minimale Reservierungsstärke festzulegen.

2.4.1.2 Verwaltung mittels dynamischer Zeitslots

In diesem Abschnitt soll zunächst ein Verfahren skizziert werden, um die quadratische Abhängigkeit der Laufzeit von der Anzahl zu bearbeitender Reservierungen zu vermeiden. Eine genaue Betrachtung des Verfahrens zeigt, dass das grundlegende Problem die redundante Betrachtung der Konfigurationen darstellt. Für ein festes Zeitintervall, in dem mehrere Reservierungsanfragen bearbeitet werden, muss wiederholt die Menge von überschnittenen Reservierungen sowie deren benötigte Ressourcen bestimmt werden. Dies kann vermieden werden, indem eine zusätzliche Datenstruktur zur Ressourcenverwaltung eingesetzt wird. Die Idee dabei ist es, die im Netz belegten Ressourcen bereits kumuliert und nach Zeit sortiert zu verwalten.

Um dies umzusetzen wird das Book-ahead Intervall, also das Zeitintervall in dem Reservierungen geplant werden können, in eine Menge von Subintervallen, die Zeitslots, unterteilt. Die Unterteilung des Book-ahead Intervalls in Slots erfolgt an allen Zeitpunkten an denen eine angenommene Reservierung beginnt oder endet. Damit ist die Ressourcenauslastung innerhalb eines Slots konstant.

Eine Anpassung der Menge von Zeitslots ist notwendig, wenn eine bestehende Reservierung geändert, gelöscht oder eine neue Reservierungsanfrage angenommen wird. Bei der Bearbeitung einer neuen Reservierungsanfrage sowie bei der Änderung des Start- oder Endzeitpunkt einer bereits angenommenen Reservierung müssen im Allgemeinen bestehende Zeitslots geteilt bzw. können existierende verbunden werden. Ersteres ist der Fall, wenn Start- oder Endzeit nicht auf Rand eines bestehenden Intervalls fällt, also im Inneren des solchen liegt. Liegt zum Beispiel die Startzeit \( t_{\text{start}} \) im Inneren eines Intervalls \([t_1, t_2]\), so wird dieses in zwei Intervalle \([t_1; t_{\text{start}}]\) und \([t_{\text{start}}; t_2]\) geteilt. Die Teilung des Slots ist hier notwendig, da sich die Ressourcenauslastung über dem betrachteten Intervall ändert. Analog kann für den Endzeitpunkt \( t_{\text{end}} \) der Reservierungsanfrage verfahren werden. Das Verbinden bestehender Zeitslots mit gleicher Ressourcenauslastung ist nicht zwingend notwendig, reduziert aber die Zahl zu verwaltender Slots. Abbildung 5 zeigt die Datenstruktur zu einer existierenden Menge von Reservierungen.
Abbildung 5: Abbildung von Reservierungen auf Zeitslots dynamischer Länge.

Ein Zeitslot enthält neben dem Start- und Endzeitpunkt Informationen über die kumulierte Auslastung der Netzressourcen für jeden einzelnen Link. Wird die Menge der Zeitslots sortiert verwaltet, ist die explizite Verwaltung der Endzeit und der Länge nicht notwendig.

Wird eine neue Reservierungsanfrage durch das System angenommen wird die Menge der Zeitslots zunächst so angepasst, dass Start- und Endzeitpunkt auf den Rand von Intervallen fallen. Dann werden für jeden Link im Netzwerk die maximal reservierten Ressourcen bestimmt. Mit Hilfe der Topologieinformationen können hieraus die verfügbaren Ressourcen für alle Links bestimmt werden. Diese Informationen können dann für die weitere Routenberechnung verwendet werden.

Die Verwendung von Zeitslots vermeidet eine redundante Kumulation der Ressourcenauslastungsinformationen. Trotzdem liegt immer noch eine implizite Abhängigkeit der maximalen Anzahl der zu verwaltenden Zeitslots von der Anzahl der angenommenen Reservierungsanfragen vor. Im Worst Case beginnt und endet jede durch das System angenommene Reservierung an einem anderen Start- und Endzeitpunkt. In diesem Fall müssen für jede Reservierung zwei neue Zeitslots durch das System verwaltet werden.

Das hier skizzierte Verfahren führt zwar zu einer subquadratischen Laufzeit in Bezug zur Anzahl der angenommenen Reservierungsanfragen, trotzdem ist die Anzahl der Zeitslots, die für die Bearbeitung einer Reservierungsanfrage betrachtet werden muss nicht nach oben beschränkt, wenn nicht einer der in Abschnitt 2.4.1.1 angedeuteten Ansätze verfolgt wird. Auch kann dieses Problem vermieden werden, indem die Anzahl von Reservierungen, die durch das System angenommen werden kann, nach oben beschränkt wird. Ein solches Verfahren ist im Allgemeinen aber nicht zu befürworten, da nicht die Anzahl annehmbarer Reservierungen sondern die zu Verfügung stehenden Ressourcen beschränkt sind. Ein anderer Ansatz, um die Anzahl zu verwaltender Zeitslots zu beschränken soll im folgenden Abschnitt 2.4.1.3 diskutiert werden.

2.4.1.3 Einführung einer Granularität

Das in Abschnitt 2.4.1.2 skizzierte Verfahren wird derzeit im ARGON Reservierungssystems eingesetzt. Um die Skalierbarkeit des Reservierungssystems auch bei einer hohen Anzahl an-
genommener Reservierungen sicherzustellen, ist es zu empfehlen in der überarbeiteten Version des Reservierungssystems eine minimale Granularität für die Länge der Zeitslots einzuführen. Dieses Verfahren stellt sicher, dass die Anzahl maximal zu verwaltender Zeitslots nicht von der Anzahl angenommener Reservierungen abhängt.

Kernidee des Verfahrens ist es, nicht die Zeitslots an die Start- und Endzeitpunkte der angenommenen Reservierungen, sondern die Reservierungen an die Zeitslots anzupassen. Hierzu wird zunächst über eine Granularität und einen festen Startzeitpunkt eine diskrete Menge möglicher Intervallgrenzen definiert. Die Länge eines Zeitslots kann damit nur dem ganzzahligen Vielfachen der gewählten Granularität entsprechen. Reservierungsanfragen r, die nicht an einer dieser Zeitpunkte starten oder enden werden auf eine Reservierungsanfrage r' abgebildet, deren Start- und Endzeitpunkt auf eine zulässige Intervallgrenze fallen. Untersuchungen haben gezeigt, dass eine Erweiterung der Reservierungsanfrage hierbei das sinnvollste Vorgehen zur Anpassung darstellt. Ein Beispiel für das Vorgehen findet sich in Abbildung 6.

Spätestens bei Implementierung des Verfahrens stellt sich die Frage, wie die Granularität für das Reservierungssystem gewählt werden soll. Die Erweiterung der Reservierungsanfrage auf die diskrete Menge von Zeitslotgrenzen führt zu einem Zielkonflikt zwischen Bearbeitungszeit und effizienter Ressourcenausnutzung. Dieser soll im Folgenden kurz skizziert werden.

Um die Zeit, die für das Preprocessing benötigt wird zu reduzieren, sollte auf der einen Seite eine möglichst geringe Granularität verwendet werden. Hierdurch reduziert sich die Anzahl der Zeitslots, die zur Bestimmung der maximal belegten Ressourcen betrachtet werden muss. Durch eine geringere Anzahl Zeitslots, die in der Preprocessing Phase betrachtet werden, reduziert sich auch die Zeit, die insgesamt für das Preprocessing benötigt wird.

Auf der anderen Seite führt die Verwendung einer größeren Granularität zur Einplanung und Reservierung von Ressourcen, die zur eigentlichen Umsetzung der Reservierungsanfrage nicht benötigt werden. Die unnötige Reservierung solcher Ressourcen kann zu Situationen führen, in denen eine Reservierungsanfrage abgelehnt wird, obwohl diese ohne Erwertung der bereits angenommenen Reservierungen auf die Zeitslotgrenzen umsetzbar gewesen wäre. Die Anzahl der unnötigen Überschneidungen ist dabei stark von der Granularität des Systems ab-

Abbildung 6: Abbildung von Reservierungen auf Zeitslots, deren Länge als ganzzahliges Vielfaches einer festen Granularität gegeben ist.
Skalierbarkeit des Netzwerkreservierungssystems ARGON

hängig. Insgesamt gilt, dass geringere Granularitäten zu einer steigenden Zahl unnötiger Überschneidungen führen.

Abbildung 7: Beispiel für eine unnötige Überschneidung von Reservierungen, die durch die Verwendung einer festen Granularität hervorgerufen wird.

Abbildung 7 zeigt ein Beispiel für eine solche Überschneidung. Aus diesem Grund sollte in diesem Zusammenhang eine möglichst feine Granularität, also ein kleiner Basiswert für die minimale Zeitslotlänge gewählt werden. Abhängig vom Anwendungsszenario gilt es, eine geeignete Granularität für das Reservierungssystem zu spezifizieren.

Das hier skizzierte Verfahren sollte in der überarbeiteten Version des ARGON Reservierungssystems verwendet werden. Im Gegensatz zur aktuellen Version des Reservierungssystems, welches lediglich dynamische Zeitslots verwendet hat, ist dadurch die Unabhängigkeit von der Anzahl der angenommenen Reservierungen gegeben. Dies führt dazu, dass die Bearbeitungszeit für eine Reservierungsanfrage nicht mehr vom Anwenderverhalten abhängig ist, was in der ersten Version des Systems nicht gegeben war. Insgesamt trägt dies zu einer besseren Skalierbarkeit des Reservierungssystems bei.

2.4.2 Path Computation

Die Anzahl der angenommenen Reservierungen hat keinen direkten Einfluss auf die Laufzeit der eigentlichen Routenberechnung. Betrachtet man die Bearbeitung von Advance Reservations, muss lediglich eine Konfiguration betrachtet werden. Die Abstraktion auf eine Netztopologie mit den verfügbaren Ressourcen ist bereits in der Preprocessingphase durchgeführt worden. Dadurch ist in dieser Phase lediglich die Bestimmung einer kostenminimalen Route notwendig.

Bei der Bearbeitung von Malleable Reservations liegt eine indirekte Abhängigkeit von der Anzahl eingeplanter Reservierungen vor. Mit zunehmender Anzahl angenommener Reservierungen steigt auch die Anzahl der Zeitslots, die durch das System verwaltet werden muss. Da die Zeitslots im Allgemeinen eine unterschiedliche Ressourcenauslastung haben, führt dies auch zu einer steigenden Anzahl von möglichen Kombinationen aus Start- und Endzeitpunkten, die in der Pfadbearbeitungsphase betrachtet werden müssen. Unabhängig vom Verfahren, welches zur Berücksichtigung der Konfigurationen verwendet wird steigt dadurch der Aufwand, um die kostenminimale Konfiguration, bestehend aus Startzeitpunkt, Dauer und Pfad bestimmen zu können. Insgesamt führt die Verwendung einer Granularität zu einer oberen Schranke für die Anzahl von Kombinationen, die zur Bearbeitung einer Malleable Reservation betrachtet werden müssen.

Neben der Granularität, welche zur Verwaltung der Ressourcenauslastungsinformationen verwendet wird, hängt die Anzahl zu berücksichtigender Kombinationen aus Start- und Endzeitpunkten auch von den in der Anfrage spezifizierten Parametern ab. Dies wird in Abschnitt 2.5.2 genauer betrachtet.
2.4.3 Postprocessing

Auch in der Postprocessing Phase liegt keine direkte Abhängigkeit der Laufzeit von der Anzahl eingeplanter Reservierungen vor. Wird neben den Konfigurationen der einzelnen Reservierung keine weitere Datenstruktur zur Verwaltung der Reservierungen verwendet, ist die Bearbeitungszeit für das Postprocessing von der Anzahl eingeplanter Reservierungen unabhängig.

Werden Zeitslots zur Verwaltung der Ressourcenauslastungsinformation verwendet, müssen alle Zeitslots, die durch die neu angenommene Reservierung überdeckt werden, aktualisiert werden. Diese Anzahl hängt zwar auch hier indirekt von der Anzahl angenommener Reservierungen ab, ist aber durch die Verwendung einer Granularität für die Zeitslotlänge nach oben beschränkt. Somit kann auch hier eine obere Laufzeitschranke spezifiziert werden. Durch Anpassung der Granularität kann auch diese Schranke beeinflusst werden, somit ist dies für die Skalierbarkeit des Systems nicht erheblich.

Wurde die Reservierungsanfrage durch das Reservierungssystem abgelehnt, ist kein Preprocessing notwendig. Insgesamt ist zu bemerken, dass das Postprocessing für Advance Reservations und insbesondere für Malleable Reservations nur einen sehr geringen Anteil der Gesamtlauzeit ausmacht.

2.5 Parameter einer Reservierungsanfrage

Neben den Parametern wie die Granularität des Systems oder die Länge des Book-ahead Intervalls die durch das Reservierungssystem gewählt werden können, hängt die Laufzeit die zur Bearbeitung einer Reservierungsanfrage benötigt wird auch von den in der Anfrage spezifizierten Parametern ab. Die folgenden Abschnitte sollen einen Überblick über diese Abhängigkeit liefern.

2.5.1 Preprocessing

Betrachtet man zunächst Advance Reservations, so hängt die Bearbeitungszeit im Wesentlichen von der Anzahl der Links im Netz – also von der Topologie – und den vom Anfrageintervall überdeckten Zeitslots ab. Letzteres wird durch die Länge der Reservierung bestimmt. Hierbei ist die Anzahl der überdeckten Zeitslots im Worst Case linear von der Länge der Reservierung abhängig. Dieser Fall tritt ein, wenn das gesamte Book-ahead Intervall in Timeslots minimaler Länge unterteilt ist.

Wird eine Malleable Reservation durch das Reservierungssystem bearbeitet, ergibt sich eine weitere Abhängigkeit von den in der Anfrage spezifizierten Parametern. Neben der Zeit zur Bestimmung der minimal verfügbaren Ressourcen für eine Konfiguration ist nun auch die Anzahl der Konfigurationen, für die die maximal im gesamten Zeitintervall verfügbaren Ressourcen in der Preprocessing Phase bestimmt werden müssen, von den spezifizierten Anfragesparametern abhängig. Grund hierfür ist, dass neben der Route auch der Startzeitpunkt sowie die Dauer der Reservierung variabel sind.

Betrachtet man eine Reservierungsanfrage mit fester angeforderter Bandbreite, so ergeben sich die möglichen Konfigurationen durch eine Verschiebung der Reservierungsanfrage zwischen dem frühesten Start- und dem spätesten Endzeitpunkt. Für diese Art von Reservierung führt dies zu einer im Worst Case linearen Abhängigkeit der Anzahl zu betrachtender Konfigurationen von der Länge des verwendbaren Reservierungsintervalls, welches durch $[t_{earliest};t_{latest}]$ gegeben ist. Hierbei bezeichnet $t_{earliest}$ den frühesten Startzeitpunkt und $t_{latest}$ den spätesten Endzeitpunkt für die Reservierung.

Neben dem Start- und Endzeitpunkt ist im Fall von Malleable Reservations auch die Bandbreite, welche für die Anfrage reserviert werden soll, im Allgemeinen variabel. Hier kann
ähnlich argumentiert werden: Betrachtet man einen festen Startzeitpunkt $t_{\text{start-x}}$, so hängt die Anzahl der möglichen Konfigurationen im Worst Case linear von der Länge des Intervalls $[c_{\text{min.bandwidth}}; c_{\text{max.bandwidth}}]$ ab. Hierbei definieren $c_{\text{min.bandwidth}}$ und $c_{\text{max.bandwidth}}$ die untere und obere Schranke für die zulässige Transferrate, die durch Sender und Empfänger verarbeitet werden kann. Auch in diesem Fall sichert die Verwendung eines Book-ahead Intervalls und einer Granularität für die Zeitslots eine obere Schranke für die maximale Anzahl der zu betrachtenden Konfigurationen zu. Abbildung 8 zeigt ein Beispiel für verschiedene zu betrachtende Konfigurationen für eine feste Startzeit $t_{\text{start-x}}$.

Abbildung 8: Für einen festen Startzeitpunkt $t_{\text{start-x}}$ steigt die Anzahl der zu betrachtenden Konfigurationen mit der Länge des Intervalls $[c_{\text{min.bandwidth}}; c_{\text{max.bandwidth}}]$.

### 2.5.2 Path Computation

Für die Routenberechnung ergibt sich ein ähnliches Bild wie in Abschnitt 2.4.2 beschrieben. Auch hier liegen in der aktuellen Implementierung keine direkten Abhängigkeiten von Parametern, die in der Reservierungsanfrage spezifiziert werden, vor. Gleichwohl müssen im Falle einer Malleable Reservations wieder alle Konfigurationen in die Routenberechnung einbezogen werden.

Für Reservierungen, bei denen keine Anforderungen an die Ende-Zu-Ende Verzögerung von Verbindungen gestellt werden, könnte ein Laufzeitgewinn erzielt werden, da eine Betrachtung aller Pfade nicht notwendig ist. Dieser Aspekt wird in Abschnitt 3 genauer betrachtet.

### 2.5.3 Postprocessing

Die Laufzeit, die für das Postprocessing benötigt wird, ist von den in der Anfrage spezifizierten Parametern unabhängig. Nach Bestimmung eines Pfades – im Fall von Malleable Reservations zusätzlich einer Konfiguration – werden die angefragten Ressourcen entsprechend der in der Routenberechnung bestimmten Lösung als reserviert eingeplant. Eine Betrachtung der in der Reservierung angefragten Parameter ist hier nicht notwendig.

### 2.5.4 Signalisierung

Neben Angaben über den Quell- und Zielknoten sowie die benötigten Ressourcen müssen auch Angaben zum Mapping des Traffics auf die konfigurierten Verbindungen spezifiziert werden, sofern ein Dienst auf IP Ebene verwendet wird. Andernfalls ist es dem Netz nicht möglich zu entscheiden, welcher über dem Eingangsport eingehende Verkehr auf den zur Reservierung gehörenden Tunnel gesendet werden soll. Konkret wird dies in der Reservierungsanfrage selbst oder durch eine zu einer Anfrage gehörende Bind-Anfrage umgesetzt.
Die spezifizierten Bindparameter werden durch das Reservierungssystem verwendet, um Access Control Lists (ACLs) auf den Routern des VIOLA Netzes zu konfigurieren. Diese Konfiguration erfolgt im ARGON Reservierungssystem mit Hilfe der Command Line Interfaces (CLIs) der Router.

In der verwendeten Implementation des Reservierungssystems bestehen ACLs aus Paaren von Quell- und Ziel IP-Adressen. Weitere Filterkriterien, wie beispielsweise Ports werden durch das Reservierungssystem nicht konfiguriert. Betrachtet man die Spezifikation der Bindparameter in den Anfragen, so werden diese in zwei Gruppen für Quell- und Zielknoten spezifiziert. Die Abbildung dieser Gruppen auf Paare von Quell- und Zieladressen, was für die Konfiguration der ACL notwendig ist, macht im Allgemeinen die Paarung aller Adressen notwendig. Auch wenn zusammenhängende Adressbereiche durch das Reservierungssystem erkannt und zu Bitmasken zusammengefasst werden, kann dies im Worst Case zu einer quadratischen Abhängigkeit der Anzahl zu konfigurierender Paare von der Anzahl spezifizierter Quell- und Zieladressen führen.
3 Ausblick

Der folgende Abschnitt stellt eine Reihe weiterer Ansätze vor, die im Rahmen des ARGON Reservierungssystems noch nicht implementiert oder noch nicht näher betrachtet wurden. Die hier vorgestellten Ansätze können dazu beitragen, die Skalierbarkeit des Systems weiter zu steigern.

3.1 Zeitslotbasierte Sperrung des Ressource Managers

Wie in Abschnitt 1.2.1 beschrieben können Operationen, die zu einer Verknappung der verfügbaren Ressourcen führen, nur seriell ausgeführt werden, um eine doppelte Reservierung vorhandener Ressourcen zu vermeiden. Eine vollständige oder zumindest teilweise Parallelisierung der Routenberechnung und der Reservierung der verfügbaren Ressourcen könnte allerdings die Skalierbarkeit des Systems bei steigender Zwischenankunftsrate der nicht-parallelisierbaren Anfragen erhöhen. Insbesondere wäre es dadurch möglich, die Bearbeitung von Anfragen, die an das Reservierungssystem gestellt werden, vollständig zu parallelisieren.

Ein erster Schritt in diese Richtung stellt die teilweise Parallelisierung zeitlich disjunkter Anfragen dar. Liegt zwischen zwei verschiedenen Reservierungsanfragen keine zeitliche Überdeckung vor, so können diese parallel bearbeitet werden, ohne dass die Gefahr besteht, dass Ressourcen für den gleichen Zeitabschnitt doppelt reserviert werden. Hierzu muss der Lock, der in den vorherigen Betrachtungen global für den gesamten Resource Manager definiert wurde, lokal auf den einzelnen Zeitslots umgesetzt werden.

Wird eine neue Reservierung durch das Reservierungssystem entgegengenommen, wird zunächst überprüft, ob die geschnittenen Zeitslots durch die Bearbeitung einer anderen Reserve rungsanfrage blockiert sind. Ist dies der Fall, ist eine parallele Bearbeitung der Reservierung nicht möglich. Auch hier muss zunächst abgewartet werden bis die Anfrage, deren Bearbeitung bereits begonnen hat, vollständig abgeschlossen ist. Liegt keine Überschneidung zwischen der neu zu bearbeitenden Anfrage und Anfragen, die derzeit bearbeitet werden, vor, können die Reservierungen parallel abgearbeitet werden. Dazu werden die Zeitslots, die von der neu zu bearbeitenden Anfrage überschnitten werden, gesperrt, um einen Zugriff durch folgende Anfragen zu verhindern. Ist die Bearbeitung der Anfrage abgeschlossen, wird die Sperrung der durch die Anfrage betroffenen Zeitslots aufgehoben.

Insgesamt führt dieses Verfahren zu einer weiteren Parallelisierung der Anfragesbearbeitung. Durch Verwendung dieses Verfahrens können sämtliche Anfragen, mit Ausnahme der sich zeitlich überschneidenden nicht-parallelisierbaren Anfragen, gleichzeitig bearbeitet werden. Dieses hier skizzierte Verfahren ist im ARGON Reservierungssystems derzeit noch nicht implementiert.

3.2 Reservierungen ohne Ende-Zu-Ende Verzögerungsanforderungen

Betrachtet man verschiedene Anwendungsbereiche, so gibt es eine Reihe von Szenarien, die keine Anforderungen an die Ende-Zu-Ende Verzögerung des Gesamtpfades stellen. Regelmäßige Backups von Datenbeständen stellen in diesem Zusammenhang nur ein mögliches Anwendungsszenario dar. Sofern neben der Kostenfunktion hierbei keine weitere Randbedingung mit additiver Metrik betrachtet werden muss, ist es nicht notwendig, alle verfügbaren Pfade im Netz zu betrachten.

Für diese Fälle könnte das Reservierungssystem eine zweite Pfadberechnungsroute zur Verfügung stellen, die lediglich den kostenminimalen Pfad berechnet, der keinen blockierenden Link enthält. Bei Verwendung eines solchen Ansatzes ist es nicht mehr notwendig alle Pfade im Netz zu betrachten, was die im Worst Case exponentielle Abhängigkeit der Laufzeit von der Anzahl der Links im Netz vermeiden würde. In diesem Fall wäre die Laufzeit der Pfadbe-
rechnungsphase durch $O(n \log n + m)$ beschränkt, wobei $n$ die Anzahl der Knoten und $m$ die Anzahl der Kanten im Netz repräsentiert.

Wird ein solcher Ansatz durch das Reservierungssystem implementiert, könnte durch ein geeignetes Abrechnungsmodell die Anfrage von Reservierungen ohne Anforderungen an die Ende-Zu-Ende Verzögerung belohnt werden. Dies würde die Skalierbarkeit für eine steigende Anzahl von Anfragen, die eine Routenberechnung erfordern, erhöhen.

3.2.1 Hierarchisierung des Reservierungssystems

Betrachtet man mehr als eine Metrik mit additiven Randbedingungen im Pfadberechnungsalgorithmus, so ist es notwendig alle Pfade im Netz zu betrachten. Für steigende Topologiegrößen führt dies im Allgemeinen zu Skalierungsproblemen, da die Anzahl der Pfade insbesondere für dicht vermaschte Netze exponentiell mit der Anzahl der Links wachsen kann.

Ein Ansatz, um dieses Problem zu vermeiden, stellt eine Hierarchisierung des Reservierungssystems dar. Dabei wird das Netz in verschiedene disjunkte Teile eingeteilt, die jeweils mit einem eigenen Ressourcenreservierungssystem ausgestattet sind. In diesem Zusammenhang liegt es nahe, sich an administrativen Grenzen, die zum Beispiel durch autonome Systeme (AS) definiert werden, zu orientieren.

Wird nun eine Advance Reservation Anfrage an das Reservierungssystem auf oberster Ebene gestellt, so muss dieses zunächst die Teilsysteme auf der nächst niedrigeren Ebene der hierarchischen Struktur bestimmen, die von einem möglichen Pfad durchlaufen werden müssen. Im Allgemeinen sind hier mehrere Alternativen zu berücksichtigen. Zur weiteren Bearbeitung wird die Anfrage in Teilanfragen zerlegt und an die betroffenen Teilsysteme weitergegeben, die die gleiche Prozedur für ihre Teilanfrage wiederholen. Durch Verwendung einer vergleichbaren Kostenmetrik kann bei Verfügbarkeit mehrerer Alternativen sichergestellt werden, dass das übergeordnete Reservierungssystem einen kostenoptimalen Gesamtpfad durch Festlegung der zu durchlaufenden Netzteile berechnen kann. Die genaue Berechnung des Pfades innerhalb eines solchen Netzteiles wird dann durch das jeweilige zugehörige Reservierungssystem durchgeführt, was zu einem deutlich geringeren Berechnungsaufwand für die einzelnen Systeme führt. Insbesondere reduziert sich die Anzahl der möglichen Pfade pro Subsystem deutlich.

Für die Berechnung einer Lösung für Malleable Reservations ist eine orthogonale Zerlegung der Anfrage wie im Fall von Advance Reservations nicht ohne weiteres möglich. Für diese Art von Reservierung sind leicht Fälle zu konstruieren, in denen die einfache Zerlegung der Anfrage zu nicht zueinander passenden Konfigurationen führt, die durch die einzelnen Subsysteme bestimmt werden. Ein Ansatz, um dieses Problem zu beheben, ist es, die Konfiguration durch das Reservierungssystem auf oberster Ebene festzulegen. Eine Möglichkeit besteht hier in der Weitergabe von Kostenvektoren durch die Subsysteme, um verschiedene Alternativen bewerten zu können.

Die hier skizzierten Herausforderungen zeigen deutlich, dass weitere Untersuchungen in diesem Bereich notwendig sind, wenn dieser Ansatz innerhalb eines Netzes implementiert werden soll.

3.2.2 Ansätze zur Erhöhung der Fehlerresistenz

Die Erhöhung der Ausfallsicherheit ist ein wichtiger Punkt, der bei steigender Topologiegröße und wachsender Anzahl Anfragen betrachtet werden muss. Das in Abschnitt 3.2.1 skizzierte Verfahren stellt eine Möglichkeit dar, um die Auswirkung lokaler Fehler zu beschränken. Trotzdem kann auch hier der Ausfall eines Reservierungssystems Reservierungen in bestimmten Teilen des Netzes verhindern oder zu einer Partitionierung des Netzes führen.
Generell kann das Reservierungssystem ähnlich wie andere Serveranwendungen vor Fehlern geschützt werden. Ein gängiger Ansatz stellt hierbei die Verwendung redundanter Systeme dar. Hierbei ist lediglich darauf zu achten, dass die Informationen über die im Book-ahead Intervall reservierten Ressourcen zwischen dem Produktivsystem und dem Backupsystem ausnahmslos abgeglichen sind. Andernfalls kann dies zu ähnlichen Problemen führen, die auch auftreten, wenn nicht-parallelisierbare Anfragen parallel ausgeführt werden.
4 Anhang

4.1 Referenzen

[B2.3.2] Ferdinad Hommes, Carsten Rosche, Jochen Schon, “Program documentation of the reservation system interface”, VIOLA Report B2.3.2, 31.10.2005

[B2.4.1] C. Barz, M. Pilz, W. Moll, F. Hommes, C. Rosche, J. Schon, “Specification of an End-to-End Signalling System under Consideration of existing Vendor Interfaces”, VIOLA Report B2.4.1, VIOLA, 15.08.2005

[MS01] Q. Ma and P. Steenkiste. Quality-of-Service Routing for Traffic with Performance Guarantees. In IFIP Fifth International Workshop on Quality of Service, NY, NY, pages 115–126, 1997.

4.2 Akronymen

ACL Access Control List
ARGON Allocation and Reservation in Grid-enabled Optical Networks
ARGON.UI ARGON User Interface
AS Autonomous System
CLI Command Line Interface
DFN Deutsches Forschungsnetz
JMS Java Message Service
NRPS Network Resource Provisioning System
OSPF Open Shortest Path First
VIOLA Vertically Integrated Optical Testbed for Large Applications in DFN
Master thesis
For the degree of M.Sc. Media Informatics

Deployment and evaluation of GMPLS Networks in Context of Scalability.

Authored by
Sajid Rashid Khan
Aachen University of Technology (RWTH), Aachen, Germany.

Supervised by
Mr. Ferdinand Hommes
Fraunhofer Institute for Media Communication – IMK
Sankt Augustin, Germany.

Prof. Dr. rer. nat. Otto Spaniol
Head of Department
Communication and Distributed Systems - Informatik 4
Aachen University of Technology (RWTH), Aachen, Germany.
Acknowledgements

I would like to express my deepest gratitude to my supervisor Mr. Ferdinand Hommes whose valuable suggestions, guidance, encouragement and continuous assistance enabled me to complete the thesis.

I am deeply indebted to Dr. Dirk Thissen from the Aachen University of technology (RWTH), whose help, stimulating suggestions and encouragement helped me in all the stages of research and writing of this thesis.

I am really thankful to Prof. Dr. rer. nat. Otto Spaniol for being my supervisor. His encouraging remarks, guidance and advices helped me to complete the work.

I owe a depth of gratitude to all my colleagues at Fraunhofer who helped and supported me to finish my work. Special thanks to all those friends who gave me the feeling of being at home during work in Germany.

Especially, I would like to give my special thanks to my family for the support and encouragement.
Dedication

I dedicate this thesis to my family and parents…
### Table of Contents

1. Introduction ................................................................................................................. 1
2. Introduction to technology ............................................................................................. 3
2.1 Optical Networks ....................................................................................................... 3
2.2 Multi-protocol Label Switching (MPLS) ................................................................... 5
   2.2.1 Principle of MPLS ................................................................................................. 6
   2.2.2 MPLS Protocols ................................................................................................. 7
   2.2.3 Applications of MPLS ........................................................................................... 7
2.3 Traffic Engineering .................................................................................................... 7
   2.3.1 Traffic Engineering with MPLS (MPLS TE) ........................................................ 8
2.3.2 Real life usages of MPLS-TE ................................................................................ 8
2.4 Generalized Multiprotocol Label Switching (GMPLS) ............................................. 9
   2.4.1 GMPLS Protocols ................................................................................................. 9
2.5 UNI Interface ........................................................................................................... 10
2.6 External Network to Network Interface (E-NNI) .................................................... 11
   2.6.1 E-NNI Signaling ................................................................................................. 11
   2.6.2 E-NNI Routing ................................................................................................. 12
   2.6.3 Compatibility with UNI ....................................................................................... 13
2.7 Internal Network to Node Interface (I-NNI) ............................................................ 13
2.8 UNI, E-NNI and I-NNI ............................................................................................ 13
2.9 VIOLA Project ......................................................................................................... 16
   2.9.1 Goals of VIOLA ................................................................................................. 16
   2.9.2 Testbed topology ................................................................................................. 17
2.10 InterEmulator ......................................................................................................... 18
   2.10.1 Benefits of InterEmulator .................................................................................... 19
3. System study ............................................................................................................... 20
   3.1 GMPLS enhancements ............................................................................................ 20
   3.1.1 RSVP-TE enhanced principles ............................................................................ 21
   3.1.2 OSPF-TE enhanced principles ............................................................................ 21
3.2 LSP generation process ............................................................................................. 22
3.3 Related work ............................................................................................................. 23
## Table of Contents

4 Research Motivation ........................................................................................................ 25

4.1 Introduction .............................................................................................................. 25

4.2 Research Purpose ..................................................................................................... 25

4.2.1 GMPLS network deployment and evaluation ...................................................... 26

4.2.2 Interoperability among partner equipments ......................................................... 26

4.2.3 Scalability study of GMPLS networks ................................................................. 27

4.3 Major tasks ............................................................................................................... 27

4.3.1 Network topologies and scenarios ....................................................................... 28

4.3.2 Connection of GMPLS InterEmulator to a real GMPLS network ....................... 28

4.3.3 Evaluation of real GMPLS networks .................................................................. 28

4.3.4 Deployment and evaluation of emulated GMPLS networks ............................... 29

4.3.5 Deployment and evaluation of hybrid GMPLS networks .................................... 29

4.3.6 Results derivation from similar emulated and real networks ............................... 30

4.3.7 Tests identification ............................................................................................ 30

4.3.8 Evaluation/Performance measurements ............................................................... 30

4.3.9 GMPLS scalability tests .................................................................................... 31

4.3.10 Emulated stress tests ....................................................................................... 31

4.4 Protocols selection from within GMPLS stack ......................................................... 32

5 System Design ............................................................................................................. 33

5.1 Network Architecture .............................................................................................. 33

5.1.1 VIOLA Network .................................................................................................. 34

5.1.2 VIOLA Addressing Scheme ................................................................................ 36

5.1.3 Data packets analysis ........................................................................................ 37

5.1.4 Two real devices network .................................................................................... 39

5.1.5 Network of two emulated devices ....................................................................... 42

5.1.6 Hybrid network of real and emulated devices ..................................................... 45

5.2 Problem description and solutions ........................................................................... 46

5.2.1 Background .......................................................................................................... 46

5.2.2 Real and emulated device simple connectivity .................................................... 49

5.2.3 Real and emulated device connectivity using IP tunnel ...................................... 50
# Table of Contents

5.2.4 Two real and two emulated devices connectivity ........................................... 51  
5.2.5 Connecting more than one emulated devices with one Real Device .............. 51  
5.2.6 Generate LSP between end UNI-C clients ..................................................... 53  
5.2.7 Summary of problems ................................................................................... 54  

6 Evaluation ............................................................................................................. 55  

6.1 Test methodologies ............................................................................................. 55  

6.1.1 Interoperability test methodology ................................................................. 55  
6.1.2 Performance test methodology ................................................................. 56  
6.1.3 Scalability test methodology .......................................................................... 57  

6.2 Tests and evaluation ........................................................................................... 57  

6.2.1 Tests for interoperability ................................................................................ 57  

6.1.1.1 Real-Real devices tests .............................................................................. 58  
6.1.1.2 Real-Emulated devices test ................................................................. 59  

6.2.2 Tests for device/network performance ......................................................... 64  

6.2.3 Tests for scalability ....................................................................................... 70  

6.2.4 Stress tests .................................................................................................... 72  

6.3 Conclusion ......................................................................................................... 73  

7 Conclusion and outlook ......................................................................................... 74  

7.1 Achievements ..................................................................................................... 74  

7.2 Outlook ............................................................................................................... 76  

References ............................................................................................................... 77
List of Figures

2.1 Typical MPLS network structure ................................................................. 6
2.2 GMPLS protocol stack ............................................................................... 9
2.3 UNI Interfacing .......................................................................................... 10
2.4 User end-to-end switched connection......................................................... 12
2.5 UNI, E-NNI, I-NNI .................................................................................... 14
2.6 Optical Internetwork Model ....................................................................... 15
2.7 VIOLA topology .......................................................................................... 17
3.1 OSPF three stages ...................................................................................... 22
3.2 LSP generation ........................................................................................... 23
4.1 Scalability .................................................................................................... 27
4.2 Hybrid network .......................................................................................... 29
4.3 Stress tests .................................................................................................. 31
5.1 VIOLA network – taken from VIOLA .......................................................... 35
5.2 Alcatel and InterEmulator nodes ................................................................. 39
5.3 Emulated ENNI .......................................................................................... 44
5.4 Emulated I-NNI .......................................................................................... 45
5.5 Emulated ENNI or INNI with clients ............................................................ 45
5.6 Real and Emulated ...................................................................................... 46
5.7 SDH connections and ports ....................................................................... 47
5.8 IP addresses and tunnels .......................................................................... 48
5.9 Real and emulated with or without tunnel ................................................ 50
5.10 Two real and emulated devices connectivity ............................................. 51
5.11 Real and multiple emulated ...................................................................... 52
5.12 One real and multiple emulated ................................................................. 52
5.13 Real and multiple emulated with UNI clients .......................................... 53
6.1 Real network test environment .................................................................. 58
6.2 Hybrid network test environment ............................................................... 60
## List of Tables

5.1 New addressing scheme ........................................................................................................... 36
5.2 Control Plane Interfaces ........................................................................................................ 39
5.3 Data Plane Interfaces ................................................................................................................. 40
5.4 Real device typical parameters ............................................................................................. 40
5.5 Emulated device additional typical parameters ......................................................................... 42
5.6 Old addressing scheme ............................................................................................................. 48
1 Introduction

High speed networks have become necessity of computer researchers, scientists, engineers, businessmen and layman users. The evolution of Internet and telecommunication technologies has even enhanced the requirement of stable, consistent and efficient networks. New standards are being defined and designed for network automation, flexibility and efficient bandwidth management. Deployment and evaluation of such standards in a heterogeneous environment of network equipments has become a challenge because different vendor devices either don’t support new standards or they claim such support but don’t fulfil this claim perfectly. Generalized Multi-protocol Label Switching (GMPLS) [13], Automatic Switched Transport Network (ASON) [1] and User to Network Interface (UNI) [18] are such new standards defined by IETF, ITU and OIF.

Purpose of this thesis is to deploy and evaluate GMPLS based networks along with ASON and UNI standards in a heterogeneous environment. A test bed [30] is needed for testing signalling mechanisms in heterogeneous environment of different vendor devices including Alcatel, Sycamore and Navtel InterEmulator [25]. The devices from these vendors are used for deployment and evaluation of different network topologies and scenarios.

A new approach for deployment of hybrid network based on GMPLS/ASON standards is used. In this approach, a hybrid network of GMPLS/ASON compliance real and emulated devices is deployed to construct a large network. This network is called hybrid because it contains both real and emulated network devices. Navtel InterEmulator software is used to deploy this hybrid network. Real devices, that are part of the testbed, are interconnected to emulated devices provided by InterEmulator software. This deployment is very useful because construction of real network lab at large scale is not feasible for evaluation purposes. This hybrid network can eventually be used for different tests and evaluations. Along with this hybrid network, real devices network of three Alcatel cross connects can also be utilized for evaluation study. Labelled Switched Path (LSP) generation mechanism [9] is evaluated for real and hybrid networks.

Different types of tests for evaluation of interoperability, performance and scalability of emulated, real and hybrid networks are designed. Interoperability of real and emulated devices is of immense concern. Successful interoperability between real and emulated devices leads to the deployment of large hybrid network. Eventually sixty nodes large hybrid network including two real Alcatel 1678 [4] cross connects is deployed. Network initialization and LSP generation performance measurements for small and large emulated and hybrid network are carried out. Scalability in context of network growth and LSP generation is an important

Deployment and evaluation of GMPLS Networks in context of scalability
topic of the thesis. LSP generation maximum load on a real device is also an important part of the observations.

This thesis consists of seven chapters:

First chapter is this one and provides an abstract introduction to the thesis.

Second chapter is about optical networks and related background technologies. Optical networks with basics of MPLS/GMPLS, UNI, E-NNI, I-NNI [18] are discussed. Short introduction to VIOLA project [30], which is the core testbed, and Navtel InterEmulator software, that was used to emulate networks, is given in the last subsections.

Third chapter is the system study. In the second chapter MPLS/GMPLS introduction was provided. In this chapter, enhancements GMPLS provides on MPLS, principles of RSVP-TE [22] and OSPF-TE [10], and an explanation of LSP generation mechanism is given. Work and research done by other institution on the same theme is also covered.

Fourth chapter is related to thesis motivation. Overall description of thesis focus and underlying approach is discussed. Major tasks carried out during the thesis are explained here.

Fifth chapter describes system design in detail. Existing and deployed network architectures are described along with VIOLA devices configuration and addressing schemes. After discussion of network configuration deployment of hybrid networks is discussed in detail with related problems and their achieved solutions.

Sixth chapter talks about final results of the conducted tests during the thesis. Different test methodologies with tests definitions and results are discussed in detail. These are all the tests performed over emulated, hybrid and real networks.

Seventh chapter wraps up the thesis work with a general conclusion that includes overall achievements. An outlook to future work is discussed in context of scalability.

The terms thesis and study have been used interchangeably in this writing. So any occurrence of study should be considered thesis unless specified explicitly.
2 Introduction to technology

This chapter describes optical networks and related background technology to help understand the theme of the thesis. This chapter assumes the reader has background of networking concepts, different networking and routing protocols.

Following topics are discussed in this chapter:

- Optical Networks
- MPLS
- UNI, E-NNI, I-NNI
- GMPLS
- VIOLA Project
- InterEmulator

2.1 Optical Networks

This is the age of high bandwidth and high performance networks where network devices transmit large amount of data in a very small fraction of time. This term ‘small fraction’ is relative and could be defined differently in different ages of technology. Today this small fraction is comparatively lesser than few years back. Researchers look forward to exponentially shrink this factor in future. Technological advancements are madly marching toward the day when users can utilize disk spaces of remote computers as fast and efficiently as they use local ones. Network speed is definitely one of the deciding parameters to achieve this goal. The promise of optical network is to fulfill the increasing bandwidth demand.

Optical network as perceived by Alcatel:

“Optical networks are high capacity telecommunication networks based on optical technologies and components that provide routing, grooming and restoration at the wavelength level as well as wavelength-based services.” [15]

Main motivation for optical networks is telecommunication industry where everyday innovations have changed the thinking and demands of the people and rapid networks are the ultimate requirement. In addition, with the escalating demand of high bandwidth in different
innovative applications like Internet, video conferencing, VOIP applications, virtual and augmented reality applications, and grid computing, optical network are considered to be a solution. Optical networks seem to accomplish increasing bandwidth needs of all these applications, but still standards for optical networks are under research and development.

To set standards for optical networks, Synchronous optical networks (SONET) was defined. SONET standardized line rates, coding schemes, bit-rates hierarchies and network maintenance functionalities. One main reason for SONET’s success was its scalability. [15]

Optical networks are the next steps to SONET with wavelength division multiplexing (WDM) [7]. Defined network elements and architectures provide basics of optical networks like SONET. However, they are different from SONET in the sense that they are based on wavelengths instead of fixed bit rate and frame structure.

Massive increase in network bandwidth because of WDM has opened a new challenge of faster switching within the core network. Different standards and protocols are being discussed and evaluated to provide faster switching with enhanced bandwidth in such a way they may also support conventional IP based network infrastructures.

Currently, there is a marvelous amount of interest and investment among network provider industry, network researchers and scholars to address the issues of next generation fast networks.

Internet Engineering Task Force (IETF) is concentrating on Generalized Multi-protocol Label Switching (GMPLS) which is based on Multi-protocol Label Switching (MPLS) [6]. With GMPLS, IETF intends to achieve the goal to provide switching among different kinds of networks that are based on different domains like time, wavelength, IP or fiber.

Optical Internetworking Forum (OIF) is committed to development and deployment of interoperable devices. OIF standards like User to Network Interface (UNI) and Network to Network Interface (O-NNI) provide mechanisms for non-optical network devices to communicate with optical networks and for interoperability between different domains in the optical network. OIF encourages different network providers, equipment providers, and end-users to establish and develop interoperable products and services.

International Telecommunication Unit (ITU) has published more than 2900 recommendations for operation and mutual inter-work between different service and products providers of the telecommunication industry. The ITU-T has recommended architecture for the management
of Automatically Switched Optical Networks (ASON). This architecture forms the basis of many recommendations within ITU-T.

2.2 Multi-protocol Label Switching (MPLS)

MPLS defines mechanisms [13] that are used to efficiently forward/switch packets within large networks. It’s Multi-protocol because it can be applied with any layer 3 protocol. Modern networks use it to achieve optical signaling, Quality of Service (QoS), and next generation VPN services.

Traditional IP networks are connection-less. When a packet reaches a router, it determines the next router depending on packet destination IP address and local routing table. However, MPLS uses a connection-oriented approach to forward packets based on pre-configured Labeled Switch Paths (LSPs).

A router that supports MPLS is called Label Switch Router (LSR). A router that connects non-LSR to LSR is called edge router. A router in a MPLS network that accepts packets from outside and transmits it to the local network is an ingress router. An egress router is a LSR that transmits packets from inside the MPLS network to outside world. The labels are tags assigned to the packets. The labels are assigned by the ingress router while entering the MPLS network and are removed by an egress router when leaving the network. Each LSR contains a label table which is consulted when packets pass through it; this is called Label Information Base (LIB). Forward Equivalency Class (FEC) refers to all the packets to which the same label is applied. FEC is used when different packets are needed to be sent on the same LSP.

Figure 2.1 shows typical MPLS network structure. Within the MPLS network different LSRs are connected to each other and maintain Label Information Base. Customer Premises Equipment (CPE) may or may not belong to MPLS based network on customer site but CPE devices can communicate to ingress and egress routers. Faster switching within the MPLS core network provides efficient routing of data packets from sender CPE device to receiver CPE device.
2.2.1 Principle of MPLS

We consider Label Distribution Protocol (LDP) [24] to understand the working principle of MPLS Label binding. LDP defines the way how LSPs are created and how labels are assigned to packets being transmitted over MPLS network. LDP works on every hop in MPLS in order to create LSP from ingress to egress LSR.

To create LSP and for label distribution, LIB is maintained on every LSR. On basis of inbound interface and label in LIB, outbound interface and label are determined. Each LSR communicates with its upstream LSR to announce its own label on a particular incoming interface. From one edge LSR to another same propagation of labels follows and LIBs are populated with neighbor LSR label/interface information. When a packet arrives at a LSR it uses its LIB to decide which link to use to forward this packet and what label to apply to forward it to the next hop. The ingress LSR uses the destination address of the packet to determine LSP while other LSRs inside the network only use labels to forward the packets.

At every edge node, label corresponding to outbound interface has no value which indicates that label needs to be removed when transmitting data outside of the MPLS network. [24]


2.2.2  MPLS Protocols

To establish LSP two different kinds of protocols are used in MPLS: routing and signaling protocols.

Routing protocols distribute network topology information to help calculate LSPs. Open Shortest Path First (OSPF) or Intermediate System to Intermediate System (IS-IS) protocol is used as a routing protocol in MPLS.

Signalling protocols are used to inform LSRs about the labels and interfaces to be used for a particular LSP. Two mainly used signalling protocols are ReSource ReSerVation Protocol [23] Traffic Engineering (RSVP-TE) and LDP.

2.2.3  Applications of MPLS

MPLS applications can be considered to gain further understanding of MPLS. Following are given few applications of MPLS: [14]

- MPLS has been considered as IP based traffic engineering technology which maps actual traffic efficiency to available resource. It can be used with existing IP based infrastructures, and carriers can improve their operational efficiency by using it.

- MPLS can work as forwarding mechanism for services based on IP, Frame Relay, ATM and Ethernet. Layer 2 transport is considered as a new application of MPLS.

- MPLS networks provide additional Class of Service (CoS) and Quality of Service (QoS) slogans to existing network infrastructures.

2.3  Traffic Engineering

Considering a network usage there are two things to understand: network engineering and traffic engineering [29].

Network engineering refers to the concept of developing a network that suits the traffic of a corporate or business. Developing network involves selection and accumulation of routers, switches and other hardware devices to fulfill traffic demands. Definition of suitable topology plays a vital role to fulfill traffic needs of a particular network.
Traffic engineering refers to the idea of manipulating the traffic in such a way that traffic fits to a given network. This is a very difficult task to engineer traffic and normally can not be achieved hundred percent successfully [11].

Rapid traffic growth and sudden flashing events like political scandals may be the reasons for shortage and demand of network bandwidth. Network underutilization like frequent links down in a network may be an addition to the bandwidth demand factor of a network. Traffic engineering is the art of moving traffic on the network so that all network links are utilized equally and flow of traffic remains smooth over the period of time on a given network.

### 2.3.1 Traffic Engineering with MPLS (MPLS TE)

MPLS TE provides a mechanism to create Label Switch Paths between end devices and data over MPLS network is forwarded on these LSPs. When building LSP for traffic, MPLS reserves bandwidth for users. This bandwidth reservation acts as a mechanism to manage network bandwidth resource and chances of network congestion seemingly decrease. As a LSP is added to the MPLS network it can find path on the network where required bandwidth is available [2].

Bandwidth reservation is made in control plan only; so if a LSR makes reservation of 20Mb and needs to send 100Mb then it can attempt to send 100Mb data over the LSP with 20 Mb reserved bandwidth. However, if some QoS policies are enforced at source then sending more than 20Mb on that LSP may not be possible.

### 2.3.2 Real life usages of MPLS-TE

Network utilization optimization, link and node failure handling, and unexpected congestion handling are considered to be main real life applications of MPLS-TE.

Network utilization optimization can be achieved by applying a so called full-mesh approach. In this method a full mesh of LSPs, knowing available bandwidth paths, is built between communicating routers and these LSPs select best available path for data transfer. This method helps avoiding network congestions.

Handling unexpected network congestion is another real life example of MPLS-TE. In this deployment strategy no pre-defined mesh of LSPs is constructed, rather normal Interior Gateway Protocol (IGP) is allowed to transfer traffic in conventional way. LSPs are constructed only after some congestion is discovered. This might be a simple approach and helps to avoid congestion in unexpected events of increased network usage.
Quick recovery of a failed link or node is another major usage of MPLS TE. Fast Reroute (FRR) component of MPLS TE is used to reroute packets to other paths of MPLS network if previously followed path goes down or a node forwarding these packet becomes dead.

### 2.4 Generalized Multiprotocol Label Switching (GMPLS)

Generalized Multiprotocol Label Switching enhances MPLS by providing a complete separation of data and control planes in network layer.

Control plane promises services like simplified network operations and management for devices that switch in packet, time, fiber and wavelength domains. Network operations and management are simplified by automating end to end node connections, providing level of QoS for high resource demanding applications. Data plane is used to transmit data between end users using control protocol which manages LSP creation between these end users.

#### 2.4.1 GMPLS Protocols

GMPLS has extended the signaling and routing protocols of MPLS. These extended protocols include RSVP-TE, Constraint based Routing – Label Distribution protocol (CR-LDP) [28], OSPF-TE, and IS-IS-TE. First two are for signaling while former two are for routing. GMPLS includes a new protocol called Link Management Protocol (LMP) [15] for link management.

LMP ensures proper health of data and control planes between two neighboring nodes in a GMPLS network. There are basically two main purposes of LMP. The first is link-channel management which is established by negotiating link parameters like keep-alive messages sending frequency and maintaining the health of the link by implementing a HELLO message flow protocol. Second is link-connectivity verification, a mechanism that takes care of physical link connectivity between any two neighboring nodes. Link-connectivity verification is performed by sending PING like test messages to nodes under test [3].

![GMPLS protocol stack](image)

*Fig 2.2: The GMPLS protocol stack [15]*
2.5 UNI Interface

With the advent of optical networks and automatic provisioning of connections using GMPLS or ASON [31], an interfacing mechanism is required for non-optical clients to connect to optical core. OIF User Network Interfacing (UNI) provides a way to support connection between such clients with optical devices. UNI is derived from GMPLS RSVP-TE specification [17] and is defined by OIF forums.

The device within a non-optical network is called UNI client (UNI-C) and requests a connection with optical device on optical core, providing UNI network (UNI-E) interface. This optical device is also referred as Optical Cross Connect (OXC). UNI-C requests connection information on an opaque optical network. In opaque optical network, UNI-C does not get any information about network topology and does not need any signaling protocols to communicate with UNI-E devices. So it makes UNI-C device interface stack implementation simple. UNI interfacing is defined in UNI 1.0 specification [18].

UNI 1.0 Release 2 incorporates ITU ASON requirements [16] for automated switching. UNI 1.0 Release 2 is not backward compatible with UNI 1.0 specification. UNI 2.0 specification provides improvements in security, real time modifications in pre-established connection, and many other features along with backward compatibility with UNI 1.0 Release 2 specification.

In the figure 2.3, two GMPLS LSR devices interfacing as UNI-C devices and providing GMPLS protocol stack functionality are connected to optical core using UNI interface.
When UNI-C 1 needs to establish a connection with UNI-C 2, it sends connection request to ingress OXC router which provide it UNI-N interface. This OXC router has network topology information and checks for available paths to UNI-C 2 and provides the best suitable path to UNI-C 1. UNI-C 1 never knows topological information of network.

2.6 External Network to Network Interface (E-NNI)

As network providers deploy Automatic Switched Transport Networks (ASON) within their networks and as it is obvious that sub-networks or domains may not be homogeneous, need for a mechanism to communicate between different domains arises. These control domains or network may be parts of the same infrastructure provide by one network provider. External Network to Network interface provides a reference point among different control domains.

Control domain is an architectural construct that provides encapsulation and information hiding for a given network domain structure. A router within a control domains is called optical cross connect (OXC) and non-optical client communicating with such cross connects is called UNI-C. Same overlay model is used, as described for UNI interface, so clients connecting to OXCs do not need information of network topology. Ingress routers know paths to proper egress routers who can deliver packets to ultimate UNI-C device.

2.6.1 E-NNI Signaling

The ITU standards G.807 and G.8080 [31] define three basic connection types according to the distribution of connection management functionality between the control and the management planes.

The following connection types have been identified: [9]

**Permanent connection (PC):** A PC is a connection type that is provisioned by the management system.

**Switched Connection (SC):** A SC is any connection that is established, as a result of a request from the end user, between connection end points using a signaling/control plane and involves the dynamic exchange of signaling information between signaling elements within the control plane(s).

**Soft Permanent Connection (SPC):** An SPC is a user-to-user connection where by the user-to-network portion of the end-to-end connection is established by the network management system as a PC. The network portion of the end-to-end connection is established as a switched
connection using the control plane. In the network portion of the connection, requests for establishment of the connection are initiated by the management plane and setup by the control plane.

Whether SC connections are needed or SPCs, E-NNI is needed to establish both kinds of connections.

Figure 2.4 shows SC and SPC path connections calls.

2.6.2 E-NNI Routing

OIF specification [16] defines an Inter-Area routing protocol called Domain to Domain Routing Protocol (DDRP). DDRP is a hierarchical routing protocol that uses OSPF-TE and nodes within one area/domain can distribute link and node information of this area. Logical devices capable of hierarchical routing, Routing Controllers (RCs), are defined that abstract local domain information to other domains.

RCs can distribute Inter-/Intra-domain links and their capabilities. In addition, they can distribute TNA addresses of end clients that want to connect to optical core. Unlike traditional IP networks where OSPF nodes are adjacent to each other, Routing Controllers are normally not adjacent in a network.


2.6.3 Compatibility with UNI

According to OIF E-NNI specification [31] its signaling should be compatible with UNI specification 1.0 Release 2. If any unknown object is received in a particular protocol, proper methods of that protocol should be used to deal with this situation.

2.7 Internal Network to Node Interface (I-NNI)

Internal Network to Node Interface (I-NNI) is a non-standard service control interface between transport networks belonging to the same domain. I-NNI is not standard so every network equipment vendor has its own specification, and devices from different vendors might not communicate over I-NNI interface.

When an optical network is deployed and if all the devices/OXCs in this network belong to same brand, I-NNI may the choice for interfacing between these devices. I-NNI is simple from configuration and management point of view as compared to E-NNI.

A UNI client can be connected to I-NNI interfacing network using UNI interface. A UNI-C client can request to an ingress router to provide connection to another client available on some egress router in the same domain. Ingress router knows the path and information of the egress router that is either directly connected to desired UNI-C client or further knows the path to client through another domain [26].

2.8 UNI, E-NNI and I-NNI

UNI provides interface from end user device that lies on non-optical network (say on IP network) to optical network devices like optical cross connects.

E-NNI is OIF standard that provides interfacing mechanism between two different domains. Routers that want to communicate with their counterparts on other domains/areas utilize E-NNI.

I-NNI is a non-standard interfacing mechanism that is utilized between nodes in one domain in an optical network. Because of its non-standard nature, devices that communicate with each other over I-NNI are normally from the same manufacturing vendor.
As exposed from figure 2.5, a client uses UNI signaling to communicate with device in an optical network. I-NNI is being used within devices of one domain. Devices that need to communicate over different domains utilize E-NNI signaling. A special router called RC (Routing Controller) is defined in each domain that communicates with other routing controllers belonging to different domains. These RC devices advertise local domain topological information to RCs of other domains and utilize same information from other RCs while setting up paths to different clients. Each client is assigned Transport Network Assigned (TNA) address which is a unique identification of client over the network.
Figure 2.6 describes UNI, E-NNI and INNI interfacing in a more simple and elaborative manner.

Optical Network

--- Fig 2.6: Optical Internetwork Model [5] ---
2.9 VIOLA Project

With an initial background of optical technology and related network/routing protocols, this part presents VIOLA project [30]. This thesis was written in context of VIOLA to provide information base related to GMPLS based networks.

VIOLA is an integrated testbed for applications and advanced network services. Different partners from industry, research laboratories, universities and DFN association have shaken hands to make it a success. Results of VIOLA may help to provide next generation science and research network (X-WIN) for Germany where technical simulations, visualizations, virtual and augmented reality applications, cluster based grid systems and other high bandwidth demanding applications need high performance network/internet infrastructure. Industrial partners will get an opportunity to test their equipment in a heterogeneous environment and will be able to improve their products in context of different desired factors like scalability and efficiency parameters.

Fraunhofer institutes (IMK and SCAI departments), Alcatel SEL AG, T-Systems International GmbH, Siemens AG, Research Centre Juelich, Stiftung caesar, University of Bonn, FH Bonn-Rhein-Sieg (University of applied sciences) and RWTH Aachen University formed a high tech consortium in the leadership of DFN-Verein to provide a solution for next generation high bandwidth research network for Germany [31].

2.9.1 Goals of VIOLA

Major goals among many others are: [31]

- Testing current signaling mechanisms in heterogeneous environment of network devices.

- Testing next generation network equipment and signaling mechanisms close to production environment.

- Development of user driven bandwidth allocation based on next generation network technologies.

- Enhancement and test of new advanced applications of virtual and augmented reality, grid and distributed computing etc.
• Preparation of X-WIN of DFN, the next generation research network of Germany, which promises to provide high performance backbone with trunk capacities of 10-40 Gbits/s and increased network intelligence/automation.

• Step by step implementation of X-WIN network with gradual enhancement of equipment and technical complexity.

• Implementation of vertical testbed of high bandwidth demanding applications, middleware tools, and optical transport technology for IP, SDH and Gigabit Ethernet.

• Co-operation with other projects of similar nature anywhere around the world.

2.9.2 Testbed topology

Figure 2.7 is a description of VIOLA testbed topology.

Initially VIOLA testbed is implemented in the region Aachen–Bonn–Cologne (State of North-Rhine-Westphalia) and eventually will be expanded to Erlangen–Nuremberg (State of

Deployment and evaluation of GMPLS Networks in context of scalability
Bavaria). Currently, the VIOLA backbone includes sites in Sankt Augustin, Jülich, Bonn, Nuremberg and Erlangen with OEO switches (Alcatel 1678, Siemens 7070, Sycamore SN 16000) connected via 10 Gb trunks. In addition, there will be 10GE switches (Riverstone 15008) in Sankt Augustin, Jülich and Bonn again connected via 10 Gb trunks.

Signaling based on MPLS/GMPLS can be tested on 10GE backbone switches and routers of various manufacturers available in VIOLA network lab. Different tests, e.g. dynamic bandwidth allocation, can be performed in VIOLA provided environment.

2.10 InterEmulator

VIOLA network lab provides limited number of cross connects and routers. For instance, only three Alcatel 1678 cross connects are currently available. In order to perform tests on large networks or test these equipments in large networks, an emulated network solution was adapted. Different real and emulated network devices are connected together to construct an emulated network.

InterEmulator [25] is a Solaris based software environment that facilitates network providers to test their network equipments in an emulated environment. The router manufacturers, stack developers, MPLS/GMPLS routing devices developers, network designers, and network researchers can get benefit of InterEmulator (IE) product.

Users can emulate network nodes like switches, routers, and cross connects in IE. These emulated nodes can be connected with each other using emulated links/interfaces and final result of this exercise in IE provides a completely emulated network. Real time emulation of IP/MPLS/GMPLS network is possible. InterEmulator is useful for companies who want to test large networks and signaling mechanisms while they don’t have access to real network resources may be because of cost.

Real routers and switching equipment can be added to any emulated topology, providing a hybrid network with both, emulated and real nodes. When an emulated device communicates with a real device, it effects in the same way two real devices could communicate.

For each node users can define signaling and routing parameters based on IETF RFCS, ITU specification, and OIF implementation recommendations [25]. Emulated node can generate control plane traffic, while components like Protocol Analyzer and Statistics viewer are provided to analyze and present traffic parameters. Users can add, delete, enable and disable LSPs and view the effect of this activity on the network.
To perform different kind of negative tests, InterEmulator provides fault injection facility. Users can inject faults on any node or link of a topology of the network. Eventually effect can be viewed. Different kind of stress tests can be applied on emulated network.

Up to 300 nodes [25] can be deployed in a MPLS/GMPLS (UNI clients, OXCs for UNI, ENNI and INNI interfaces) based environment in InterEmulator. With one centralized configuration repository, entire GMPLS/MPLS network can be distributed over multiple Solaris workstations. Solaris workstation with centralized control/repository is InterEmulator server while other distribute network nodes are InterEmulator clients.

### 2.10.1 Benefits of InterEmulator

InterEmulator provides a lot of benefits for MPLS/GMPLS based network testing. Some of these are summarized below: [25]

- Real emulation of GMPLS/MPLS networks. VIOLA GMPLS test network can be emulated with different emulated and real devices available.

- A combination of real and emulated nodes forms a large hybrid network which can be tested in the same way as an entire large real network.

- High Signaling performance testing for VIOLA GMPLS network.

- Additional nodes can be added very easily in a pre-defined network to test scalability.

- It is easy to change topologies, links, topology configurations, and protocol attributes and effect of these changes can be traced.

- Testing LSP generation and scalability testing with large number of LSP generated on a given network.

- Protocol operation and scalability testing under network failure and recovery scenarios.
3 System study

This chapter discusses the enhancements GMPLS provides on MPLS. Principles of RSVP-TE and OSPF-TE are described. In addition, LSP generation mechanism is explained. Work and research done by other institutions is part of this chapter as well.

Following sub-sections are included:

- GMPLS enhancements
  - RSVP-TE
  - OSPF-TE
- LSP generation
- Related work

3.1 GMPLS enhancements

GMPLS is the next generation implementation of MPLS. Two main areas which have been extended by GMPLS are routing and signalling. A new protocol Link Management Protocol (LMP) also has been added.

Enhanced IGP-TE (Internet Gateway Protocol with Traffic Engineering) routing, numbered and un-numbered link support, and TE-links and component with bundling support are the main enhancements in context of routing protocols.

Signalling related enhancements include Forwarding Adjacencies (FAs) and protection-restoration support [8]. GMPLS extends the functionality of MPLS by allowing different kind of LSP generation options for variety of optical devices. Bi-directional LSP are allowed to be created between two end devices.

LMP is one of the major enhancements that GMPLS provides. LMP is used for GMPLS link connectivity verification and control channel management. The links between adjacent nodes consists of data and control links. A Hello packet is used to check the aliveness of the control link. Summarization of multiple data links along with failure localization procedures are defined as part of LMP.
3.1.1 RSVP-TE enhanced principles

Generalized Multi-Protocol Label Switching (GMPLS) extends MPLS to include support for Layer-2 (L2SC), Time-Division Multiplex (TDM), Lambda Switch Capable (LSC), and Fiber Switch Capable (FSC) interfaces [21]. GMPLS-based recovery uses control plane mechanisms (i.e., signalling, routing, link management mechanisms) to support data plane fault recovery. Note that the analogous (data plane) fault detection mechanisms are required to be present in support of the control plane mechanisms [17].

RSVP-TE has been extended from MPLS and provides path setup facility in core GMPLS based networks. Switch type, encoding type and generalized protocol id have been added in GMPLS by extending label request object of MPLS [19]. Bi-directional LSP have been allowed. In a bi-directional LSP two unidirectional LSPs are created from source to destination nodes and vice versa. For bi-directional LSPs generation upstream label object has been defined [9].

3.1.2 OSPF-TE enhanced principles

Discussed here are general principles that are used in the evaluation study in the thesis.

Routers using OSPF goes through three stages

- Neighbour discovery
- Database synchronization
- Routes calculation

In the local area adjacencies between neighbouring nodes are created. HELLO protocol is used to discover neighbours and select border router (BR) for the local area. Once the neighbours are known and BR is elected, next step is to synchronize routers databases. Link state attributes for each router are sent from one to another. Each router propagates this information to its neighbouring router. This link state information is propagated to BR. BR communicates to BRs of other areas and sends local are link state information to them. After all the BRs and local routers link state databases are synchronized OSPF enters into the third stage of routes calculation. SPF algorithm is used to calculate shortest path to all destination routers. This information of optimized routes is kept into forwarding tables of routers. After the calculation of all these optimized routes to all destinations, OSPF node comes to FULL STATE.
Figure 3.1 gives an overview of all three stages of OSPF.

![OSPF Stages Diagram]

**3.2 LSP generation process**

This section discusses LSP generation process that confirms to standard UNI 1.0 release 2 [18]. To understand the connection establishment process four network nodes can be considered. Source UNI-C is the client that wants to initiate connection call to destination UNI-C. Source UNI-N is the optical cross connects that is directly connected to source UNI-C. Destination UNI-N is another optical cross connected to destination UNI-C on the other end.

To initiate a connection source UNI-C sends a PATH message to source UNI-N. This PATH message is a path request message to destination UNI-C. Source UNI-N sends back ACK for this PATH message and forwards this PATH request to destination UNI-N cross connect. Destination UNI-N finally forwards this PATH request to destination UNI-C. Destination UNI-C client acknowledges this PATH message with RESV message. This indicates that path is reserved on destination UNI-C for source UNI-C. Destination UNI-N sends back ACK message to acknowledge this RESV message. After this event source UNI-N sends RESV message to source UNI-C. Source UNI-C responds with RESVCONF message, this is a confirmation response for path establishment from source UNI-C. Source UNI-N sends ACK for this RESVCONF message. At this point of time RESVCONF message is sent from
destination UNI-N to destination UNI-C. The destination UNI-C acknowledges this message with ACK message. A path has been established from source UNI-C to destination UNI-C.

Figure 3.2 demonstrates above described process.

![Figure 3.2: LSP Generation](image)

Above paragraphs provide an overview of the successful LSP generation procedure. In case of any problems PATHERR message is sent from problematic node to request initiator.

### 3.3 Related work

Many other research networks execute important tests and preparations of future network technology.

CANARIE is the Canadian organization working on research and development for high speed research networks. Its mission is to accelerate Canada's advanced Internet development and use by facilitating the widespread adoption of faster, more efficient networks and by enabling the next generation of advanced products, applications and services to run on them.
GÉANT2 network is a European project to achieve high-performance, state-of-the-art network infrastructure for European Union. UKERNA, CESNET or PIONIER are among other European projects that aim on research and development for high speed network.

All the above mentioned projects are or have dealt with similar or related research questions addressed in the thesis. Next chapter provides an overview and motivation of the study conducted during the thesis.
Chapter 4 – Research Motivation

4 Research Motivation

This chapter articulates basic idea and motivation behind the thesis. Overall description of study focus and underlying approach is discussed.

Following are the main topics illustrated in the chapter:

- Detailed description of the study
- Research Purpose
- Major tasks: using InterEmulator and real networks
- Protocols selection from within GMPLS stack

4.1 Introduction

In the study GMPLS based network deployment and evaluation has been researched in context of VIOLA project [11] that stands with the main goals to test new generation high-tech network equipment and signalling mechanisms in heterogeneous environment of router and switching devices.

To provide a flexible and high capacity research network for the future, equipment from Industrial partners is under research in VIOLA to gather information base. Conducted study provides basic information about implementation of dynamic provisioning of bandwidth, GMPLS network scalability, and selected available network equipment interoperability testing.

4.2 Research Purpose

As stated earlier, main emphasis of study is on Generalized Multi-Protocol Label Switching (GMPLS) networks deployment, scalability of GMPLS networks, and network equipment interoperability tests. Following points precisely describe main purposes of this study and these points are explained in more detail in next coming sections:

- Deployment related issues of Generalized Multi-Protocol Label Switching networks with VIOLA partners’ equipments.
- Study of interoperability of partner equipment in a GMPLS network.
- Scalability study of GMPLS networks.
4.2.1 GMPLS network deployment and evaluation

As described earlier a small network of Alcatel 1678 routers is operational among Bonn, Juelich, and Sankt Augustin VOILA sites. This pre-deployed GMPLS network was used to exercise different identified [evaluation chapter] tests.

As a next step, an entirely emulated network has to be established which is similar as real network defined above. Similar tests [evaluation chapter] need to be exercised on this emulated network to take the same readings as gathered from real network. This emulated network will eventually enlarge in order to perform tests on different topologies and size of networks.

Finally, a network needed to be constructed that emulates large number of InterEmulated nodes and integrates available real nodes. It constituted a hybrid network with a few real and large numbers of emulated nodes. Different identified tests [evaluation chapter] needed to be performed on this hybrid GMPLS network.

4.2.2 Interoperability among partner equipments

Sankt Augustin, Juelich and Bonn sites make backbone of VIOLA testbed. As described earlier, each site maintains Alcatel 1678, Siemens hiT 7070, Sycamore SN 16000 cross connects/routers that are interconnected over Gigabit trunks. These cross connects support GMPLS label switching.

In order to test GMPLS based interoperability among these vendors, a small GMPLS network of these vendor devices needed to be deployed keeping in mind that if these devices communicate with each other then existing network can be extended, in future, with additional cross connects from the same vendors.

GMPLS is an emerging standard of IETF and different network equipment providers have started to offer GMPLS functionality in their devices. In this initial phase of GMPLS technology it is natural to have issues and problems while interconnecting different vendor specific devices in the same network.

Results of interconnection of GMPLS enabled devices from different vendors available in VIOLA need to be explored. If these devices communicate over GMPLS as expected then evaluating the stability and performance of this multi-vendor GMPLS based network is part of study.
In addition, study deal with interfacing this GMPLS based network with non-optical network based devices, testing the interoperability with such devices and eventually finding network performance meters on basis of different number of end-to-end devices.

### 4.2.3 Scalability study of GMPLS networks

One significant fraction of this study is to conduct tests to check scalability of a GMPLS based networks. Different kinds of tests need to be defined in order to check how a network under study behaved in different situations? These tests could be performed on pure real network and/or emulated networks.

Starting with a small network of two emulated devices and adding a number of UNI clients that request LSPs over the optical core, a large network of these emulated routers could be tested for parameters like maximum number of clients supported, maximum number of LSPs supported by a specific vendor, and time needed to provide LSPs for newly added clients. Similar readings could be taken by applying negative testing. For example, requesting a LSP path from one client to another before and after the removal of an optical cross connect from the operational GMPLS network.

### 4.3 Major tasks

In order to accomplish overall objectives of this study, following main concrete steps/phases were identified:

- Network topologies and scenarios
- Connection of GMPLS InterEmulator to a real GMPLS network
- Evaluation of real GMPLS networks.
• Deployment and evaluation of emulated GMPLS networks.

• Deployment and evaluation of large real/emulated GMPLS networks.

• Result derivation from similar emulated and real networks

• Tests identification

• Evaluation/Performance measurements

• Scalability testing

• Emulated stress tests

4.3.1 Network topologies and scenarios

This task consists of definition and deployment of different network topologies. These topologies should eventually be deployed by integration of emulated and real network nodes. Different scenarios based on identified tests need to be designed and executed.

4.3.2 Connection of GMPLS InterEmulator to a real GMPLS network

This task is important to construct a hybrid network of emulated and real devices. As described earlier, InterEmulator provides mechanism to connect emulated devices with real network devices to construct hybrid network. This part of study is conducted for Alcatel 1678 cross connects. Successful connectivity and operation is an immense requirement. At the start of study it was not known whether Alcatel devices would operate with this virtual/emulated environment or not. This hybrid network might be of immense use for findings of the study.

4.3.3 Evaluation of real GMPLS networks

A simple GMPLS network of three Alcatel 1678 cross connect nodes, among Sankt Augustin, Bonn, Juelich, is established and this study focuses on this GMPLS network. Different tests like functional or operational tests, performance tests, capacity or scalability tests, and dynamic path provisioning tests for real devices and network are part of the study.

LSP generation tests from Alcatel provided real proxy UNI clients is part of the study. In addition to that, InterEmulator was of great use to include emulated clients and routers in this small real optical fabric.
In order to test UNI interface, E-NNI interfaces need to be configured and evaluated in devices belonging to this real GMPLS network.

4.3.4 **Deployment and evaluation of emulated GMPLS networks**

Using Navtel InterEmulator same network topology need to be configured and deployed as it is available in real VIOLA lab environment. Different network parameters like link configuration attributes, OSPF attributes, RSVP attributes etc should be configured exactly same in both cases.

In InterEmulator, definition and execution of Emulated Test Sessions is planned. Moreover, InterEmulator based emulated applications (EAPP) need to be designed and developed that could be used in emulated test sessions. UNI interface testing is supported in InterEmulator so clients need to be emulated and tested with GMPLS based emulated network. As stated earlier, due to non-availability of Alcatel UNI proxy client, InterEmulator based UNI clients could extensively be used.

4.3.5 **Deployment and evaluation of hybrid GMPLS networks**

Once both emulated and real GMPLS networks are in operational condition, next step is to integrate them and make a hybrid network. This hybrid network could incrementally be enlarged to test the effect of newly added devices to network. So tests like functional or operational, performance, capacity or scalability, and dynamic path provisioning can be performed in this hybrid environment as well.

**Fig 4.2: Hybrid network**
The results of this experiment can be used to predict the behaviour of similar real GMPLS network. Further, these results may be helpful to get readings of different network performance parameters for real devices.

Testing UNI interface with real or emulated devices is integral part of this component of study.

**4.3.6 Results derivation from similar emulated and real networks**

If possible, derivation of a generalized model of test findings can be established. There are different situations, as defined above, where we can apply our identified tests [evaluation chapter] to GMPLS network. The results from tests that are executed on real network can be used as reference and comparison of these results with findings of emulated network can provide a pattern or model. If such a model exists then it can be used to derive results for large networks which may be defined in InterEmulator. This model will make easy to test large emulated networks in InterEmulator environment and map the results to real networks.

In case, such a model is not established then findings of larger scalable networks in InterEmulator can be recorded. These findings may provide an understanding of general behaviour of large real networks.

**4.3.7 Tests identification**

Having established real, emulated and hybrid GMPLS network, an important part of this study is to identify the tests that can be performed on these networks. Different parameters of scalability, performance, functionality and interoperability are the guiding factors to identify and design tests.

In case of real network, manual tests for basic readings were performed. In InterEmulator extensive use of emulated test sessions environment could be beneficial. As provided by InterEmulator, it is easy to write automated scripts to perform different tests. CLI API of InterEmulator was utilized to write such automated tests. These automated tests were designed keeping in mind the parameters under study.

**4.3.8 Evaluation/Performance measurements**

As tests are identified [Evaluation chapter], evaluation/performance measurements of real, emulated, and hybrid networks deployed in InterEmulator can be carried out.
Performance measurements of different actions (identified tests) on deployed network topologies need to be performed. These events/actions include Path Creation, Path deletion, Backup paths, adding/deleting new hops etc.

Different numbers of UNI clients can be added to a deployed topology to increase the load of control traffic on the network, LSPs are created/deleted to generate different load levels to check the performance of network.

4.3.9 GMPLS scalability tests

Different scalability tests needed to be identified [12]. These tests can be performed in InterEmulator provided environment or manually on real network devices.

Similar to performance measurements, clients can incrementally be added to certain GMPLS based network topology and effect of network growth on Alcatel devices and emulated nodes need to be observed. This finding gives an overall behavioural picture of a scalable network that grows with time.

In addition, different factors of OSPF and RSVP scalability need to be identified and evaluated.

4.3.10 Emulated stress tests

Stress tests environments are supported by InterEmulator. It was very useful for scalability and performance tests. Automated stress tests need to be designed and implemented using InterEmulator environment to add large number of UNI clients in a network or by adding/deleting LSPs.

Fig 4.3: Stress Tests
4.4 Protocols selection from within GMPLS stack

Three major categorizations could be observed in order to understand which protocols contribute in GMPLS and eventually selected of protocols [29] from within this GMPLS stack was carried out for tests and evaluations.

- **Routing protocols:** OSPF-TE and IS-IS-TE are two basic routing protocols that are used for advertisement of resource availability including bandwidth and protection type. Auto-discovery of network topology is another major task of routing protocol suit. IS-IS-TE is that connectionless network protocol (CLNP) is used to transmit information when IS-IS-TE is used. In case of OSPF-TE, IP is used. This study intended to utilize OSPF-TE in deploying GMPLS test environment.

- **Signalling protocols:** RSVP-TE or CR-LDP can be used in GMPLS protocol. Purpose of these protocols is management of traffic engineered paths (LSPs). E.g. connection (LSP) generation, deletion etc. RSVP-TE protocol was used in order to establish GMPLS network.

- **Link Management Protocol:** LMP is used for GMPLS link connectivity verification and control channel management. This is the only choice for link management in GMPLS, so eventually was used.

Now after making clear understanding of research objectives and major tasks identification, overall theme of the thesis should be clear. Forthcoming chapters deal with procedures that were followed to achieve these objectives and working on these tasks.
5 System Design

This chapter provides detailed description of VIOLA network design and architecture [30]. VIOLA real network is discussed with different nodes, their connectivity and related configuration parameters. In addition, emulated network comprising real and emulated nodes is described in detail with different design and deployment problems and their corresponding solutions. Only basic network design and architecture is discussed in this chapter. Detailed deployment descriptions and strategies are discussed in evaluation chapter.

In brief, this chapter talks about:

- Network architecture
- VIOLA network and real devices
- Real and emulated networks
- Problems and their solutions

5.1 Network Architecture

An essential step in this study was to understand VIOLA requirements and previously developed real network architecture. On the basis of this network, an emulated network was designed and deployed to perform tests.

This section defines different basic architectures that are deployed for testing purposes. Following main topics are discussed in this section:

- VIOLA network
- VIOLA addressing scheme
- Data packets analysis
- Network of two real devices
- Network of two emulated devices
- Hybrid network of real and emulated devices

Deployment and evaluation of GMPLS Networks in context of scalability
5.1.1 VIOLA Network

Sites at Fraunhofer IMK Sankt Augustin (FhG IMK), University of Bonn (Uni Bonn), and research centre Juelich (FZJ) each host a Alcatel 1678 SDH switch/router, one Riverstone 16008 10GE, and one Alcatel 7750 SR 10 GE cross connect. In addition, one SDH Siemens hiT 7070 cross connect is hosted at each FhG IMK and Uni Bonn. TSI Nuernberg and University Erlangen host Sycamore SN 16000 and Siemens hiT 7070 cross connects respectively.

Riverstone 16008 devices are interconnected via 10GE with each other on all the three locations. Same connectivity pattern is true for all Alcatel 7750 SR routers on the network. Alcatel 1678 at each location connects via 10 Gbps SDH to similar device in other two locations. Siemens hiT 7070 SDH devices are connected with each other in between FhG IMK and Uni Bonn sites. Siemens hiT 7070, in Uni Erlangen, is connected to Sycamore SN 16000 machine in TSI Nuernberg which is further connected to SDH Alcatel 1678 in FhG IMK. All network devices provide bandwidth connectivity to external grid computing application users.

This study concentrated on SDH Alcatel 1678 cross connects hosted in FhG IMK, Uni Bonn and FZJ Juelich. These devices were eventually connected to emulated devices in InterEmulator to provide a large emulated network. Interpretability tests were performed between Sycamore SN 16000 device in TSI Nuernberg and Alcatel 1678 in FhG IMK.
Figure 5.1 provides the overall layout of the VIOLA network.

Another x86 Intel machine called piano is also hosted in FhG IMK where InterEmulator software is deployed. All emulated networks are deployed and handled on this machine.
5.1.2 VIOLA Addressing Scheme

Alcatel 1678 cross connects in FhG IMK, Uni Bonn and FZJ are named as a78sa, a78bn, and a78ju respectively. Each site has two redundant machines, service and congi, running all the times. One of them is active at a time and other stands back as passive entity. To connect to active management (EM) node IP addresses 192.168.20.10, 192.168.20.40 or 192.168.20.70 need to be used for FhG IMK, Uni Bonn, and FZJ sites respectively.

Machines service or congi can directly be connected using corresponding IP addresses. Alcatel provides Generalised MPLS Routing Engine (GMRE) that is GMPLS-based control plane implementation software. GMRE nodes can be connected through 192.168.21.19, 192.168.21.79, and 192.168.21.1 IP addresses for FhG IMK, FZJ and Uni Bonn respectively.

Table 5.1 summaries these IP addresses:

| Alcatel 1678 | a78sa            | a78bn            | a78ju            |
|--------------|------------------|------------------|------------------|
| IP Address of active EM | 192.168.20.10 | 192.168.20.40 | 192.168.20.70 |
| IP Address of active DCR   | 192.168.20.11 | 192.168.20.41 | 192.168.20.71 |
| Service EM IP address      | 192.168.20.12 | 192.168.20.42 | 192.168.20.72 |
| Service DCR IP address     | 192.168.20.13 | 192.168.20.43 | 192.168.20.73 |
| Congi EM IP address        | 192.168.20.14 | 192.168.20.44 | 192.168.20.74 |
| Congi DCR IP address       | 192.168.20.15 | 192.168.20.45 | 192.168.20.75 |
| GMRE                      | 192.168.21.19 | 192.168.21.1 | 192.168.21.79 |

Table 5.1: New addressing scheme
Each UNI client connecting to an optical network needs unique identification. OIF-UNI provides a new and distinct address space for UNI clients[14]. So a unique address is assigned to each UNI client, this address is referred as Transport Network Assigned (TNA) address. Each cross connect in VIOLA network is assigned a unique TNA address. Normally following addressing scheme is used to assign TNA addresses

\[
\text{place.device.slot.port}
\]

Where:

**place**: site where device exists

**device**: TNA is assigned to this device

**slot**: slot on this device

**port**: port on this device

TNA address for VIOLA devices are assigned in the same way. Here are some examples of assigned TNA addresses:

1 16 7 1 s16nu (SDH Sycamore 16000 in Nuernberg)
8 70 10 1 s70er (Siemens hiT 7070 in Uni Erlangen)
4 70 3 4 s78bn (Siemens hiT 7070 in Uni Bonn)
3 16 2 2 a78sa (Alcatel 1678 in FhG IMK)
3 16 x y Navtel (Navtel machine in FhG IMK)

### 5.1.3 Data packets analysis

A critical exercise in entire study was analysis of the data packets transmitted among different optical devices in deployed network. It was crucial to read and check different RSVP and OSPF packets. Without in-depth analysis of these packets it was almost impossible to successfully deploy operational network and conduct desired tests.

Three possible methods were employed to read and understand transmitted data packets.

First, statistics viewer and signalling/routing analyzers [25] provided by InterEmulator were used extensively. But these tools have few limitations:
• It was only possible to check packets that have been sent or received by InterEmulator. So the packets that never reached or packets that were un-recognizable by InterEmulator could not be seen here.

• Sometimes InterEmulator sent packets to wrong virtual interfaces so they were never transmitted over network but these tools showed packet successfully sent.

• InterEmulator shows packets in GUI and not everything desired information is visible sometimes.

• ‘snoop command’ packets transmitted between two real devices are never shown in InterEmulator.

Analysis Tools provided by InterEmulator were very useful especially when tracing messages between different emulated devices or when during successful communication between emulated and real devices.

Second option was to use packet sniffer (snoop command) on Solaris box (piano) where InterEmulator was working. It is very flexible and powerful but without any GUI. snoop was used extensively, especially in cases when InterEmulator was erroneously configured and was using wrong sub-interfaces.

Thirdly, ethereal was configured on piano that can read packets within VIOLA optical network. It was the best option with flexible and powerful GUI interface. However, signalling or routing packets sent from/to some emulated to/from other emulated devices could not be analyzed here.
5.1.4 Two real devices network

As described earlier a GMPLS network of three Alcatel 1678 devices exists among Bonn, IMK and Juelich nodes. To work with this network, unit study of each node and its connectivity to other neighbouring nodes was performed.

In order to get indispensable tests execution, a network of two devices from this setup was utilized. Bonn and IMK nodes were considered for this network study.

Figure 5.2 shows the network inter-connectivity of Alcatel 1678 (A 1678) machines. Alcatel cross connect in FhG IMK (a78sa) is connected to similar cross connects in Uni Bonn (a78bn) and FZJ Juelich (a78ju) over 10 Gbps interface. InterEmulator hosting emulated devices is also interconnected via 10 Gbps interface. These network nodes are connected via E-NNI interface making three different domain networks.

The GMRE [3] on each Alcatel 1678 cross connect manages the UNI/NNI protocol and allows dynamic path setup from external clients. GMRE CLI (Command Line Interface) was used to configure and manage cross connects.

Table 5.2 shows control plane interfaces that need to be configured while setting up an Alcatel cross connect using GMRE CLI [4].

| Control Plane Interfaces                  |                                                                 |
|------------------------------------------|-----------------------------------------------------------------|
| IP Tunnel interface                     | It is used to specify the control plane IP neighbour connectivity.
| RSVP interface                          | It is used to specify UNI/NNI neighbour and uses IP tunnel for remote node address. |
DPR interface

It is used to specify Data Plane Routing neighbour and also uses IP tunnel interface for remote node address.

LMP interface

It is used to specify LMP neighbour, uses IP tunnel interface for remote node address.

Table 5.2: Control Plane Interfaces [4]

Table 5.3 shows the control plane interfaces that need to be configured while setting up an Alcatel cross connect using GMRE CLI [13].

| Data Plane Interfaces |
|-----------------------|
| Databearer interface  | It specifies transmission capability with local and remote interface addresses. |
| TE-Link interface     | It specifies TE parameters for databearer interface. |

Table 5.3: Data Plane Interfaces [4]

Once all control and data plane interfaces are configured and their administrative state is set to value ‘UP’, this cross connect is ready to communicate with other devices.

Table 5.4 shows a list of typical parameters used to configure Alcatel cross connect in FhG IMK.

| Name          | Value         | Description                          | Interface   |
|---------------|---------------|--------------------------------------|-------------|
| ifName        | XYZ_Augustin_Navtel | Name of this interface               | all         |
| adminState    | UP            | Interface is UP and working          | All         |
| linkType      | E-NNI         | Can be E_NNI/NNI/UNI                 | databearer  |
| RemoteIfAddress | 130204 (e.g.) | Remote Interface id                 | teLink      |
| Parameter               | Value     | Traffic Engineering Metric | Metric Type |
|-------------------------|-----------|----------------------------|-------------|
| Temetric                | 1         | Traffic Engineering Metric | teLink      |
| MinimumServiceType      | VC4       | It can be chosen from one of: VC4, VC4_4c, VC4_16c, VC4_64c | teLink      |
| LinkType                | E-NNI     | It can be chosen one of: E-NNI, I-NNI, UNI | teLink      |
| OSPFType                | E-NNI     | It can be chosen one of: E-NNI, I-NNI | dpr         |
| RSVPType                | E-NNI     | E-NNI, I-NNI, UNI | rsivo       |
| HelloEnabled            | TRUE      | TRUE or FALSE | rsvo        |
| Encapsulation           | NOENCAPSULATION | NOENCAPSULATION or IPMINIMAL | Iptunnel    |

Table 5.4: Real device typical parameters
5.1.5 Network of two emulated devices

Main concern while designing this emulated network was the establishment of same parameters and factors as defined in real devices network described in subsection 5.1.1. Exactly the same real network parameters were used while defining network nodes (cross connects), links, and UNI interfaces. These emulated network nodes were inter-connected via corresponding emulated links. This small network can be considered as a replica of real network defined above.

Table 5.5 shows a list of typical parameters used to configure emulated device in InterEmulator. Parameters defined in Table 5.4 were exactly the same on emulated device as well.

| Name                  | Value   | Description                                                                 |
|-----------------------|---------|-----------------------------------------------------------------------------|
| Interface Cost       | 1       | The cost of sending a data packet on the interface, expressed in link state metric (used with the routing protocol to determine the optimal data path). |
| Bandwidth (Gb)       | 10      | The interface bandwidth in megabits per second that is used by the control plane protocol. |
| Address Type         | IPv4    | IPv4 or IPv6                                                                |
| Protection Type      | UN_PROTECTED | The Link Protection Type represents the protection capability that exists for a link. This information is used by the CSPF (Constrained Shortest Path First) algorithm to set up LSPs with the appropriate protection characteristics. |
| Label Range Type     | SDH     | It can be one of SDH, SONET, Wavelength, ATM                                |
| Switching Type       | Time_Division_Multiple_C | It can take one value from many available, |
## Chapter 5 – System Design

Deployment and evaluation of GMPLS Networks in context of scalability

| Parameter                          | Value   | Description                                                                 |
|------------------------------------|---------|-----------------------------------------------------------------------------|
| Minimum Reservable Bandwidth (mbps)| 1       | Minimum bandwidth that can be reserved.                                     |
| LabelRange MaxIn (OSPF)            | 1048576 | Maximum incoming label range for the component.                             |
| LabelRange MaxOut (OSPF)           | 1048576 | Maximum outgoing label range for the component.                             |
| HELLO Interval (OSPF)              | 10 seconds | OSPF exchanges HELLO packets. It specifies the length of time, in seconds, between the HELLO packets that router sends on the OSPF interface. |
| DEAD Interval (OSPF)               | 40 seconds | The time, in seconds, without receiving HELLO packets before the neighbour declares the router down. |
| HELLO Failure Interval (RSVP)      | 5 seconds | HELLO Failure detection interval.                                            |
| Refresh Interval (RSVP)            | 30 seconds | Refresh interval for PATH/RESV messages.                                    |

Table 5.5: Emulated device additional typical parameters

E-NNI and I-NNI interfaces were separately configured with defined OSPF, RSVP and other link parameters. Templates for links, workstations, emulated/real routers, and topologies can be defined in InterEmulator. These templates can be used as blueprint while defining new network nodes. Templates with parameters shown in table 5.5 were defined for deployment of desired GMPLS network.
Chapter 5 – System Design

Emulated network behaviour was monitored and analyzed by using different statistics viewer tools and signalling/routing analyzer tools available in InterEmulator. Tools available in InterEmulator for generation and deletion of LSPs were used.

In order to configure E-NNI interface between two emulated devices, both devices needed to be configured as RC controllers. This leads to inter-domain links configuration between these two devices. Inter-Domain configuration requires special TE link configuration between these two devices with local, remote device names, ids, interfaces and bandwidth (bits per second) for this link.

Figure 5.3 presents two emulated devices network with E-NNI interfaces.

Figure 5.4 presents two emulated devices network with I-NNI interfaces.

Configuration of I-NNI interface between two emulated devices is relatively simple if we compare it with E-NNI interface definition. It does not involve any RC definition and corresponding special TE link configuration.

Figure 5.4 presents two emulated devices network with I-NNI interfaces.
One UNI client on each emulated cross connect is added to E-NNI or I-NNI based network topology. LSP path are created between these two clients from either end.

5.1.6 Hybrid network of real and emulated devices

This type of network is established to check the behaviour of a real device using InterEmulator environment. One Real and one emulated device are added to make an emulated network. Emulated device in this setup is configured having compatible OSPF and RSVP parameters with real Alcatel device.
Addition of UNI clients on both ends of this two nodes hybrid network topology are needed to test LSP path generation/deletion.

5.2 Problem description and solutions

In order to understand different network architectures or structures it is very important to recognize fundamental complexities and issues that needed to be solved during emulated network deployment. While working with InterEmulator it seemed straightforward to establish simple network as required in this study but it was a complicated task. In this section few problems in conjunction with their accomplished solutions are examines.

5.2.1 Background

As known from earlier chapters, this study uses pre-deployed real GMPLS network of Alcatel 1678 cross connects. This section presents an overview of historical background of VIOLA network architecture and major changes in addressing schemes that directly relates to the thesis. Moreover, these changes affected the emulated network deployment and evaluation work.

In the start of study, real devices in VIOLA network had loopback addresses and corresponding actual addresses. Figure 5.7 provides a layout of network:
As shown in the figure 5.7, the piano machine was connected to a78sa. L in the figure 5.7 indicates loopback address and M stands for management or actual IP address.

All these devices were connected to each other via IP tunnels. Configuration of IP tunnel on piano could be achieved in two ways. First, by defining a tunnel manually in Solaris using `ifconfig` command. Second, by defining tunnel in InterEmulator environment using its GUI. Defining IP tunnel in InterEmulator required MAC addresses of source and destination machines. Defining IP tunnels at Alcatel side was comparatively trivial task.

Figure 5.8 shows IP tunnel connectivity:
Connectivity between Alcatel and InterEmulator devices was very complex due to these IP tunnels. It was observed that such configurations and connectivity was never fully tested from either side.

Table 5.6 provides complete listing of old IP addressing scheme being used by different devices in the network. All the tests in the forthcoming sections use these names and addresses.

| Device Name | Loopback address | Actual address | Description |
|-------------|-----------------|----------------|-------------|
| Piano       | 194.94.199.225  | 192.168.3.253  | Name and addresses of first emulated cross connect |
| piano2      | 194.94.199.226  | 192.168.3.253  | Name and addresses of first emulated cross connect |
| piano3      | 194.94.199.227  | 192.168.3.253  | Name and addresses of first emulated cross connect |
Following subsections present some of these problems that were faced during deployment of emulated networks in InterEmulator.

### 5.2.2 Real and emulated device simple connectivity

A simple topology with one real *a78sa* and one emulated *piano* device needed to be developed and loaded in InterEmulator. Table 5.6 gives addressing details of both devices.

Both devices are connected through ENNI interface. *piano* is properly configured as RC. Both RSVP and OSPF are made enabled but on loading the topology neither RSVP signalling HELLO messages nor OSPF routing HELLO messages work. InterEmulator shows that OSPF messages are being sent from *piano* to *a78sa*. These messages can be confirmed from Ethereal as well. In addition to that Ethereal shows OSPF and RSVP messages transmitted from *a78sa* to *piano* but InterEmulator does not show any message being received from outside.

Solution to this problem requires few undocumented features configurations on InterEmulator side. These configurations were not mentioned in any of InterEmulator manuals. For instance, an environment variable named ‘MNE_ADD_INTERFACE_TODEVICE’ needed to be set to ZERO in order to configure InterEmulator to accept packets on proper interface. After this setting RSVP and OSPF messaging started working.

There were two scenarios for the deployment of this structure. First, all devices were configured with old addressing scheme as defined in section 5.2.1. In addition to

|     | 192.168.255.3 | 192.168.3.41 | Name and addresses of Alcatel cross connect in IMK |
|-----|---------------|--------------|--------------------------------------------------|
| a78sa | 192.168.255.4 | 192.168.4.41 | Name and addresses of Alcatel cross connect in Bonn |
| a78ju | 192.168.255.5 | 192.168.5.41 | Name and addresses of Alcatel cross connect in Juelich |
| s16nu | 194.94.199.220 | 192.168.1.9 | Name and addresses of Sycamore cross connect in Nuernberg |

Table 5.6: Old addressing scheme
configuration defined in above paragraph, IP tunnels were defined between both devices involved in this topology. IP tunnels scenario is described in the next section 5.2.3. Second, all devices were configured with new addressing scheme as defined in section 5.1.2. Since new addressing scheme was deployed to avoid IP tunnelling problem, so in this scenario no IP tunnel was defined in this topology.

5.2.3 Real and emulated device connectivity using IP tunnel

A simple network of emulated and real device was needed where connectivity was under IP Tunnel. Need of this structure arrived because of old IP addressing scheme section 5.2.1 of VIOLA network.

Loading such a topology in InterEmulator leads to problem of packets loss either on InterEmulator or on Alcatel side. From Ethereal trace it was clear that RSVP and OSPF messages were being sent from a78sa to piano but in InterEmulator no messages were shown as received. Ethereal did never show any messages from piano to a78sa. On the other hand, InterEmulator showed signalling and routing messages being sent to a78sa real device.

To find out solution, proper IP tunnel was set up on Alcatel device from a78sa to piano, similarly, manual IP tunnel was created on piano machine from piano to a78sa. This manual IP tunnel creation effort did not work because InterEmulator did not use any manually created tunnels. InterEmulator provides a mechanism to add ip-in-ip tunnels using source destination addresses and corresponding MAC addresses. It seemed defining an ip-in-ip tunnel in InterEmulator did not work if it did not include MAC addresses of both ends of tunnel. After defining proper, as required by InterEmulator, ip-in-ip tunnel from piano to a78sa, all messages from/to piano and a78sa started working. Signalling and routing messages transmission could be witnessed from Ethereal and/or InterEmulator signalling/routing analyzer. Figure 5.9 gives a look of topology.

There was no need of IP tunnels after the definition of new addressing scheme defined in section 5.1.2.
5.2.4 Two real and two emulated devices connectivity

A simple topology with two real and two emulated devices needed to be developed and loaded in InterEmulator. Real devices were a78sa and a78bn. Emulated devices were piano and piano2. Table 5.6 gives addressing details of all devices.

Both emulated devices, piano and piano2, are connected through ENNI interfaces to real devices, a78sa and a78bn, respectively. All the devices are properly configured as RC. Both RSVP and OSPF are made enabled but on loading the topology neither RSVP signalling HELLO messages nor OSPF routing HELLO messages work. InterEmulator shows that OSPF messages are being sent from piano to a78sa and piano2 to a78bn. These messages can be confirmed from Ethereal as well. In addition to that Ethereal shows OSPF and RSVP messages transmitted from a78sa to piano and a78bn to piano2 but InterEmulator does not show any message being received from outside.

Solution to this problem requires few configurations on InterEmulator side. For each emulated device defined in InterEmulator a sub-interface is associated and need to be defined. This sub-interface can either be added/configured manually on Solaris shell prompt or such sub-interface can be added in InterEmulator GUI. In case of piano and piano2, ge0:1 and ge0:2 sub-interfaces were added. In case of more emulated devices to be added in the topology, more sub-interfaces (say ge0:3, ge0:4 ...) need to be defined. After this sub-interface is added and configured as transmission media, RSVP and OSPF messaging started working.

In case of old addressing scheme IP tunnels were used but after definition of new addressing scheme, as defined in section 5.1.2, no IP tunnels were used.

5.2.5 Connecting more than one emulated devices with one Real Device

After the connection between one real and one emulated devices worked, it was next step to connect more than one emulated devices defined in InterEmulator with real device. Two devices piano and piano2 were defined in a topology and connected to real device Alcatel 1678 in Sankt Augustin i.e. a78sa. This topology structure was initially tested before the
definition of new addressing scheme so old addressing scheme as defined in section 5.2.1 was used.

It seemed obvious that connection from piano2 to a78sa should work in the similar manner as it was working from piano to a78sa. In current working environment only one interface was available on Alcatel 1678 cross connect with IP address 192.168.3.41. InterEmulator requires more than one interface on a real device that needs to be connected to multiple emulated devices. Effectively, as many interfaces are required on real device as emulated devices need to be connected to it.

Adding a real hub in this topology could solve the problem but there was an issue of IP tunnel in this case. As in InterEmulator, defining an IP tunnel requires both destination and local address with corresponding MAC addresses and no address was available for HUB side. So it was concluded that solution is not possible in presence of IP tunnels. After definition of new addressing scheme, all IP tunnels were removed and this topology worked without any problem.

Figure 5.12 shows topology with hub and without tunnel:
5.2.6 Generate LSP between end UNI-C clients

Once all devices were configured in a hybrid network environment, it was time to test LSP creation between end UNI-C clients. For this purpose two network structures were used. First, network defined in section 5.2.4 with two emulated and two real devices was used. Two UNI-C clients were added on both ends of the network. One UNI-C client was connected to piano and other with piano2. All RSVP signalling and OSPF routing worked properly. Proper connectivity could be checked either from ethereal or InterEmulator. But any request for LSP generation from UNIC1 to UNIC2 or vice versa always failed. After a deep investigation and research it was observed that Alcatel had implemented latest RSVP-TE standard as defined in RFC3473 [22], RFC3477 [19] and internet draft [33]. InterEmulator did not implement complete standards defined in these documents.

To understand the error situation consider following scenario. When a RSVP PATH message is generated from UNIC1 indicating UNIC2 as destination address, this message was recognized and forwarded by piano to a78sa. a78sa accepted and forwarded this message to piano2. The emulated node piano2 generated PATH ERROR with error code 24 (ROUTING ERROR) and value 1 (Bad EXPLICIT_ROUTE object [33]). As a result this error propagated to backward path until UNIC1 reached. Actually InterEmulator was not recognizing a sub-object named RRO object [33] sent by Alcatel.

So eventually, a request was sent to both vendors. Alcatel was asked to disable this particular sub-object, if possible. While Navtel was asked to either implement this feature or provide some remedy to pass through this problem. Finally Alcatel disabled this sub-object on its routers and problem was solved.
5.2.7 Summary of problems

Deployment of the hybrid network was the most complex and time consuming part of the thesis. InterEmulator is a nice tool for such network emulations but it has many configuration problems for real devices connected to the emulated in complex scenarios. These scenarios include defining hybrid network with IP tunnels and configuring a real device with multiple emulated devices. There were partially undocumented features in InterEmulator that were crucial for network deployment. It was unnecessarily needed to directly communicate with Navtel customer support and technical staff for these features.

Interoperability with Alcatel devices was never tested by Navtel in depth as was needed in the thesis. There were problems in context of GMPLS/ASON and UNI standard compliance between real and emulated devices. Latest RSVP-TE draft features [33] were implemented by Alcatel but Navtel did not worked on these. This eventually leaded to problems and wastage of time.

The thesis was started with VIOLA old address scheme and configurations. Sycamore and InterEmulator devices could not communicate properly in that environment, mainly because of IP tunnels. Eventually new addressing scheme was introduced and IP tunnels were removed away. It was a real problem to work with different vendors devices because these devices were never tested in the areas where this thesis concentrated. There were problems from device/InterEmulator limitations to RSVP-TE standard implementations.

At this place, all concrete tasks have been defined and design of networks with corresponding problems has been investigated. The next chapter provides complete results along with deployed topologies.
Chapter 6 – Evaluation

6 Evaluation

This chapter provides final results of all the tests conducted during the study. Before proceeding to test results, certain test methodologies are defined that were used as basis during testing. Test definition and results section proceeds methodologies section to give a depiction of all the tests.

Following main topic compose this chapter:

- Test methodologies
- Test definitions and results

6.1 Test methodologies

Test methodologies define steps, procedures and approaches kept in mind while designing GMPLS network and related tests. On the basis of these methodologies final tests were devised and performed.

Test methodologies, used in study, can be discussed in context of two major concepts:

- UNI Testing
- E-NNI Testing

All the methodologies used were defined having these two major standards in mind. All the forthcoming sub-sections discuss both of them in detail.

6.1.1 Interoperability test methodology

This is functional testing of real and emulated devices. There were two main mechanisms to test interoperability. One was between real to real device test and other was real device to emulated device interoperability [27].

Interoperability between two different vendor real devices was tested; whether such two devices connect or not using GMPLS protocols, O-UNI and E-NNI interfaces?

To test real and emulated devices interoperability, one real device was configured to communicate with emulated devices. This scenario results in hybrid network construction. Real and emulated devices were connected through E-NNI interface while emulated devices were inter-connected using E-NNI or I-NNI interfaces.
Once a hybrid network is established, further optical cross connects and UNI-C clients are added to this network. For this purpose UNI-C, UNI-N/E-NNI devices are emulated and configured. UNI-C emulated interface is required to test into GMPLS core of the network and UNI-N/E-NNI interface is required to test an E-NNI device. UNI-N device provides network end of NNI interface.

Sycamore and Alcatel 1678 devices were tested for interoperability. Alcatel devices were tested with InterEmulator based emulated devices.

Different operational tests include OSPF, RSVP-TE parameters configuration and protocol conformity. OSPF and RSVP HELLO messages are traced in order to check proper communication of two nodes at any particular time.

LSP generation from one end device to other is part of tests. Different network vendor devices are connected in a hybrid network topology and emulated UNI-C clients are added to this topology. LSPs are generated between these UNI-C clients.

### 6.1.2 Performance test methodology

Concentration is made over two aspects in this particular case:

- Network initialization
- LSP generation

Network initialization testing means time taken by two or more nodes of fully emulated and hybrid style network. Different OSPF and RSVP messages and protocol initialization is tracked.

Performance testing is required to check behaviour of a real device when a number of LSP calls are initiated from UNI-C clients. This number can be very high to test performance level of real device in this particular case.

A specific number of LSP generated over a given time interval is tested. How many LSPs are allowed in minimum time (say one second) on a particular vendor device (Alcatel 1678)? It is expected that number of LSP allowed depends on maximum allocated bandwidth. In performance testing, foremost interest is to check how efficiently these LSPs are generated?

Different link connections with different parameters (e.g. bandwidth) are used to check device and network performance. Mix of E-NNI and I-NNI interfaces are used to develop test topologies. Different numbers of UNI-C clients are connected to this optical network to increase the load.
6.1.3 Scalability test methodology

This testing methodology initiates the need to deploy large networks and different performance tests are conducted.

A small network is started with two emulated devices only with interface either E-NNI or I-NNI. This network, either fully emulated or hybrid, is allowed to be grown by adding different emulated cross connect devices or UNI-C clients. In case of hybrid network approach, this initial network may consist of only one real and one emulated cross connect connected via E-NNI interface. Eventually, large number of emulated cross connects or UNI-C clients are added within this network. This growing network becomes the source of scalability findings for network initialization and LSP generation etc.

A hybrid network of approximately sixty nodes was constructed with E-NNI and I-NNI interfaces. Different scalability tests were performed: including maximum number of LSPs allowed on real device and maximum concurrent LSPs per minute. As said before these tests were conducted by gradually incrementing cross connects and UNI clients. Effect of cross connect increment on LSP generation was observed. Similar tests were conducted with increasing UNI clients to check the LSP generation time.

6.2 Tests and evaluation

This section provides the definition of all tests, their execution environments and corresponding results. All the methodologies described above are applied for all these tests. In proceeding subsections different tests are described in context of defined methodologies.

6.2.1 Tests for interoperability

Different tests were conducted after defining domains. Alcatel 1678 and Sycamore SN16000 were used to conduct the tests. Each device formed a new domain. So two Alcatel device connected to each other formed two domains. Different vendor devices were interconnected. For easy understanding three domains – ‘Domain A’ for Alcatel device, ‘Domain S’ for Sycamore device and ‘Domain E’ for emulated device – were defined.

Following sections described definition and configurations of the tests conducted. Corresponding results of these tests were included with some conclusive remarks. These are abstract description of tests.

- Basic routing functionality test
- Basic signalling functionality test
Chapter 6 – Evaluation

- LSP connection establishment test
- Graceful deletion of LSP connection initiated from source NNI device
- Graceful deletion of LSP connection initiated from destination NNI device

6.1.1.1 Real-Real devices tests

Domain A and Domain S devices were connected using E-NNI interface. Figure 6.1 shows real domains involved in real devices tests.

![Real network test environment](image)

E-NNI interface was used between the domains and a protocol analyzer machine hosting ethereal was used to monitor the network traffic.

**T1 – Test name and description:** Basic routing functionality test between Sycamore and Alcatel 1678 cross connects.

In case of routing tests, OSPF protocol messages were tested and verified according to specifications [10] and [20]. Links parameters were configured as defined in Table 5.4. IP control channel network was monitored to ensure that proper OSPF messages were transmitted. Databases description messages, Link state update message and other related messages were part of this monitoring. On the neighbouring nodes OSPF database was verified for correct updates, this check included database synchronization of routing controller of both domains.

**Test configuration and deployment:**
**Results:** OSPF connectivity could not be established between Alcatel and Sycamore devices. Following facts need to be considered before understanding results:

One IP tunnel between Alcatel and Sycamore devices was established and was working properly. This tunnel was required because of old IP scheme of VIOLA. OSPF multicast packets were tunnelled over the defined IP tunnel.

Test resulted in finding that devices of these vendors don’t communicate because of the following reasons:

- Alcatel does not accept OSPF multicast packets that are tunnelled.
- Sycamore does not accept OSPF packets sent over a tunnel. This is because loopback address and Ethernet IP addresses differ.

No further tests were conducted because of basic routing messages failure. These tests were postponed to January 2006. Alcatel and Sycamore need to perform these tasks before January 2006. Alcatel and Sycamore agree on:

- Alcatel should change loopback addresses and IP tunnels are removed.
- Sycamore will investigate a solution for generating unicast OSPF messages.
- Sycamore will investigate the problem of receiving OSPF packets from an IP tunnel.
- Alcatel will investigate OSPF multicast messages over IP tunnel.

### 6.1.1.2 Real-Emulated devices test

Domain A and domain E devices were connected via E-NNI interface. Domain A was a set of Alcatel real cross connects devices. Devices within domain A were connected via E-NNI interface. Domain E was set of all emulated devices and these devices were inter-connected either via I-NNI or E-NNI. These devices constructed hybrid network. Initially only one device was added inside one domain but eventually emulated domain was incremented up to 60 devices and one more real device was added to domain A.

Figure 6.2 shows domains involved in this hybrid network.
Protocol analyzer ‘ethereal’ was used on piano and on an external machine on the network. Different types of emulated network architecture were examined for interoperability tests. These architectures are defined below:

- One real and one emulated device.
- One real and multiple emulated devices.
- Two real and two emulated devices.

**Basic routing functionality test**

OSPF protocol messages were tested and verified according to specification [10] for basic routing. OSPF messages were monitored over ethereal and InterEmulator analyzer to ensure that proper HELLO, database description and different link state messages were transmitted. On the neighbouring nodes OSPF database was verified for correct database updates, this check included database synchronization of routing controller of both domains.

Following tests (T2-T4) were performed in context of basic routing functionality.

**T2 – Test name and description:** One real and one emulated device

**Test configuration and deployment:** A small topology of one real and one emulated node was deployed in InterEmulator. Real node used was a78sa in IMK Fhg. Both devices were connected using E-NNI interface. Each device was configured as routing controller making a domain. Different parameters shown in Table 5.4 and Table 5.5 were configured in real and emulated devices respectively. Further information about the topology is given in section 5.2.2 and IP addresses information is available from Table 5.1
Results: OSPF HELLO messages were being transmitted properly between the neighbouring OSPF nodes in the area. Different Database description (DD), Link state message (LSU, LSA) were observed to be properly transmitted between real and emulated nodes. As a result, Link State Database was observed to be synchronized in between these nodes.

T3 – Test name and description: One real and two emulated devices.

Test configuration and deployment: A small topology of one real and one emulated node was deployed in InterEmulator. Real node used was a78sa in IMK FhG. Two emulated devices were connected through hub, defined in InterEmulator. These emulated devices were connected to real device via E-NNI interface. Each device was configured as routing controller making a different domain. Different parameters shown in Table 5.4 and Table 5.5 were configured in real and emulated devices respectively. Further information about the topology is given in section 5.2.5 and IP addresses information is available from Table 5.1

Results: OSPF HELLO messages were being transmitted properly between the neighbouring OSPF nodes in the area. Different Database description (DD), Link state message (LSU, LSA) were observed to be properly transmitted between real and emulated nodes. As a result, Link State Database was observed to be synchronized in between these nodes.

T4 – Test name and description: Two real and two emulated devices.

Test configuration and deployment: A small topology of two real and two emulated nodes was deployed in InterEmulator. Real devices used were a78sa in IMK FhG and a78bn in Uni Bonn as defined in Table 5.1. Each emulated device was connected to a real device via E-NNI interface. Moreover, both real devices were also interconnected via E-NNI. Each device was configured as routing controller making a different domain. Different parameters shown in Table 4.4 and Table 4.5 were configured in real and emulated devices respectively. Further information about the topology is given in section 5.2.4 and IP addresses information is available from Table 5.1

Results: OSPF HELLO messages were being transmitted properly between the neighbouring OSPF nodes in the area. Different Database description (DD), Link state message (LSU, LSA) were observed to be properly transmitted between real and emulated nodes. As a result, Link State Database was observed to be synchronized in between all these nodes.

Summary: All above three tests were successful. OSPF messages should be considered to/from every neighbouring node. OSPF neighbour column shows the status of OSPF neighbour at both a78sa (GMRE) and emulated node. LSA, LSR, LSU and DD respectively
stand for Link State Acknowledgement, Link State Request, Link State Update and Database Description messages.

**Basic RSVP signalling functionality tests**

RSVP protocol messages were tested and verified according to specifications [22] and [18] for RSVP signalling. Major thing observed was RSVP HELLO messages. RSVP messages were monitored over ethereal and InterEmulator analyzer to ensure that proper HELLO messages were being exchanged between RSVP neighbouring nodes. T5 was performed in this context.

**T5 – Test name and description:** One real and one emulated device

**Test configuration and deployment:** A small topology of one real and one emulated (piano) node was deployed in InterEmulator. Real node used was a78sa in IMK Fhg. Both devices were connected using E-NNI interface. Each device was configured as routing controller making a domain. Different parameters shown in Table 5.4 and Table 5.5 were configured in real and emulated devices respectively. Further information about the topology is given in section 5.2.2 and IP addresses information is available from Table 5.1

**Results:** RSVP HELLO messages were being transmitted properly between the neighbouring nodes. In GMRE on Alcatel real machine, emulated node was indicated as RSVP neighbour. InterEmulator showed a78sa as being RSVP neighbour.

**Summary:** RSVP HELLO messages were successfully observed in both the directions.

**LSP Generation/deletion tests**

In these tests (T6 to T8), end to end LSP generation was tested using UNI-C and UNI-N interfaces. Different emulated devices (say piano and piano2) and real devices (say a78sa and a78bn) constructed the emulated hybrid network. LSP was generated from one UNI-C client to other. RSVP PATH messages were monitored over ethereal. This LSP generation test confirms UNI specification [18].

**T6 – Test name and description:** In a network of two emulated and one real device.

**Test configuration and deployment:** An emulated topology consisting of two emulated cross connects (piano and piano2) and one Alcatel real device a78sa was deployed. These emulated nodes were connected to real device via E-NNI interface. Two emulated clients were connected to emulated cross connects. Different parameters shown in Table 5.4 and Table 5.5 were configured in real and emulated devices respectively. Further information
about the topology is given in section 5.2.5 and IP addresses information is available from Table 5.1.

**Results:** Test was successful. All PATH, RESV, RESVCONF, PATHERR and ACK messages were observed to be successful.

**T7 – Test name and description:** In a network of two emulated and two real devices.

**Test configuration and deployment:**

An emulated topology consisting of two emulated cross connects and two Alcatel real devices, a78sa and a78bn, was deployed. One emulated node was connected to a78sa while other was connected to a78bn via E-NNI interface. Two emulated clients were connected to emulated cross connects. Different parameters shown in Table 5.4 and Table 5.5 were configured in real and emulated devices respectively. Further information about the topology is given in section 5.2.4 and IP addresses information is available from Table 5.1.

**Results:** Test was successful. All PATH, RESV, RESVCONF, PATHERR and ACK messages were observed to be successful. UNI client on either end could successfully establish LSPs to destination UNI clients. Messages were observed to be passed successfully among piano, a78sa, a78bn, piano2 and corresponding UNI clients.

**T8 – Test name and description:** In a network of two real and multiple emulated devices.

**Test configuration and deployment:**

An emulated topology consisting of two Alcatel real devices (say a78sa and a78bn) and multiple emulated cross connects (say piano and piano2) was deployed. One emulated node piano was connected to a78sa while another piano2 to a78bn via E-NNI interface. More emulated cross connects were connected to these two emulated nodes. Different domains were constructed and with each domain I-NNI interface was used. Multiple emulated clients were connected to emulated cross connects. Different parameters shown in Table 5.4 and Table 5.5 were configured in real and emulated devices respectively. Further information about the topology is given in section 5.2.4 and IP addresses information is available from Table 5.1.

**Results:** Test was successful. All PATH, RESV, RESVCONF, PATHERR and ACK messages were observed to be successful. UNI client on either end could successfully establish LSPs to destination UNI clients. Messages were observed to be passed successfully among piano, a78sa, piano2 and corresponding UNI clients.
Summary: All the tests related to LSP generation from T6 to T8 were successful and UNI client devices could create LSPs.

6.2.2 Tests for device/network performance

All related subsections describe the tests that were performed in context of performance. Performance evaluation of real device under different network loads and scenarios was of one concern during the tests. Another major aspect was performance evaluation of complete network during initialization or load times. LSP generation in different scenarios was another focus of this test section.

Different scenarios that are discussed in coming subsections are:

- Two emulated devices
- Two real devices
- One emulated and one real device network
- Hybrid network of 50 devices with E-NNI and I-NNI interfaces
- Hybrid network with UNI-C clients

Network initialization tests

These tests (T9 to T11) are needed to check the time required to initialize a pure emulated, fully real and hybrid network of different emulated and real devices. Time of initialization is observed by InterEmulator and ethereal RSVP/OSPF messages traces. It was the time taken by two devices to exchange RSVP HELLO [9] message and the time taken by two routing controllers to synchronize local link-state databases [10]. Time was noted when a device received/sent RSVP HELLO message i.e. it entered init state and send/received HELLO packet from other device. For OSPF time was started by taking into account time stamp of first HELLO message sent/received and then tracking DD, LSR, LSU and last LSA packet from remote node. This time information will provide total time taken by reaching RSVP and OSPF nodes on ‘full state’.

Total time taken to reach RSVP full state was calculated by:

\[
TTTR = FHR - FHS
\]

Where  TTTR: Total Time Taken in RSVP full state

FHR: First HELLO Received
Chapter 6 – Evaluation

FHS: First HELLO Sent

Total time taken to reach OSPF full state was calculated by:

\[
TTTO = LLR - FHS \quad \text{(6.2)}
\]

Where  TTTO: Total OSPF Time Taken in OSPF full state

FHS: First HELLO Sent

LLR: Last LSA Received

T9 – Test name and description: Two emulated devices network initialization

Test configuration and deployment:

Two emulated cross connects were connected via E-NNI interface making two domains. RSVP HELLO messages and OSPF routing was observed according to [9] and [10] documents respectively.

Results:

RSVP: TTTR as calculated by equation 6.1: 1 to 2 milliseconds. This was expected behaviour because all emulated devices worked in the same application context (InterEmulator).

OSPF: TTTO as calculated by equation 6.2: 7-8 seconds. Time was normal for all stages of OSPF protocol.

T10 – Test name and description: One emulated and one real device network initialization

Test configuration and deployment:

Two cross connects, one emulated (piano) and other real in IMK (a78sa), are connected via E-NNI interface making two domains. RSVP HELLO messages and OSPF routing are observed according to specification [9] respectively. Network initialization was confirmed by InterEmulator that shows status of all the devices and by ethereal traces.

Results:

RSVP: TTTR as calculated by equation 6.1: 10-20 seconds. This was expected behaviour because of 30 seconds HELLO interval time.

OSPF: TTTO as calculated by equation 6.2: 12-15 seconds. Time was normal for all stages of OSPF protocol.

T11 – Test name and description: Two real devices network initialization
Test configuration and deployment:

Two real Alcatel cross connects, in IMK and Bonn, are connected via E-NNI interface making two domains. RSVP HELLO messages and OSPF routing are observed according to specification [9] respectively.

Results:

**RSVP:** TTTR as calculated by equation 6.1: 10-20 seconds. This was expected behaviour because of 30 seconds HELLO interval time.

**OSPF:** TTTO as calculated by equation 6.2: 6-10 seconds. Time was normal for all stages of OSPF protocol for the real devices.

**OSPF-TE link state database synchronization time and size tests**

These tests (T12 to T15) were carried out to check the time required for synchronization of the link state database and total size taken for OSPF-TE link state database for a large emulated network with I-NNI and E-NNI interfaces. Time needed to synchronize link state database and corresponding database size among three routing controllers was observed. This database size was observed by recording the size of all OSPF messages send from one routing controller to other. When these routing controllers’ link state databases are in sync it means all device within one domain have already been synchronized their databases.

**T12 – Test name and description:** OSPF-TE link state database synchronization time for emulated network

Test configuration and deployment:

In this test three domains (E-NNI interfaces) were deployed. Each domain contained more than ten cross connects. All the devices within one domain were connected via I-NNI interface. Three routing controllers were defined. Routing controllers were connected via E-NNI interface. To observe the OSPF messages time stamps InterEmulator messages trace was used.

Results: Total time was 6-7 seconds.

**T13 – Test name and description:** OSPF-TE link state database synchronization time for the hybrid network

Test configuration and deployment:
In this test four domains (E-NNI interfaces) were deployed. Two real cross connects, in IMK and Bonn, were connected to each other via E-NNI and one emulated cross connect was connected to each real node via E-NNI interface, thus making four domains. Two emulated domains were constructed on either emulated side of real device. All the devices in this each intra-domain arrangement were connected via I-NNI interfaces. To observe the OSPF messages time stamps ethereal and InterEmulator messages trace was used.

**Results:** Total time was 10-15 seconds.

**T14 – Test name and description:** OSPF-TE link state database synchronization time for the real network

*Test configuration and deployment:*

In this test three domains (E-NNI interfaces) were deployed. Each domain contained more than 10 cross connects. All the devices within one domain were connected via I-NNI interface. Three routing controllers were defined. Routing controllers were connected via E-NNI interface. To observe the OSPF messages sizes InterEmulator messages trace was used.

**Results:** Total time was 6-10 seconds.

**T15 – Test name and description:** OSPF-TE link state database size for the hybrid network

*Test configuration and deployment:*

In this test four domains (E-NNI interfaces) were deployed. Two real cross connects, in IMK and Bonn, were connected to each other via E-NNI and one emulated cross connect was connected to each real node via E-NNI interface, thus making four domains. Two emulated domains were constructed on either emulated side of real device. All the devices in this each intra-domain arrangement were connected via I-NNI interfaces. To observe the OSPF messages sizes ethereal and InterEmulator messages trace was used.

**Results:** DD, LSU, LSR and LSA messages were sent. All devices within one domain sent messages to its neighbour and the neighbour propagate this information to its neighbours. Ultimately local domain information was exchanged between all devices. Each device contained OSPF link state information of other devices. Routing controller propagate this information of local domain in a summarized manner.

**Summary:** if the number of cross connects are increased in a network then initialization takes more time. This network initialization time increases linearly. Performance of a cross connect initialization depends on its memory and processing power because OSPF link state attribute database size and processing instructions.
LSPs generation time tests

These tests (T16 to T21) were performed to observe the time taken in full process of LSP generation. Time was calculated for different network scenarios as defined below. Time was calculated from first PATH message send and RESVCONF message received.

**T16 – Test name and description:** Time taken in LSPs generation in two nodes emulated network

**Test configuration and deployment:**

Two emulated nodes were configured with E-NNI node and two UNI clients were attached on either end. LSP generation request was sent from one client to other (from either end) and all messages traces were monitored using InterEmulator protocol analyzer view.

**Results:** 30-40 milliseconds

**T17 – Test name and description:** Time taken in LSPs generation in two nodes real network

**Test configuration and deployment:**

Two real devices Alcatel 1678 in IMK and Bonn were tested for LSP generation. Both were interconnected using E-NNI interfaces. LSP was generated from node from IMK to Bonn and vice versa. This test was repeated with Juelich node as well. Messages were traced using Ethereal.

**Results:** 5.1 seconds

**T18 – Test name and description:** Time taken in LSPs generation in two nodes real network with UNI clients

**Test configuration and deployment:**

Two real devices Alcatel 1678 in IMK and Bonn were tested for LSP generation. Both were interconnected using E-NNI interfaces. One real UNI proxy client from Alcatel was connect on either end. LSP was generated from node one client connected with IMK to other client connected with Bonn and vice versa. This test was repeated with Juelich node as well. Messages were traced using Ethereal.

**Results:** 100-160 milliseconds.

**T19 – Test name and description:** Time taken in LSPs generation in a small emulated network
**Test configuration and deployment:**

Six emulated nodes were configured with E-NNI interfaces and two UNI clients were attached with two cross connects. LSP generation request was sent from one client to other (from either end) and all messages traces were monitored using InterEmulator protocol analyzer view.

**Results:** 60-80 milliseconds

**T20 – Test name and description:** Time taken in LSPs generation in a small hybrid network

**Test configuration and deployment:**

Four emulated nodes were configured with two real devices. Connections between real and emulated devices were configured with E-NNI interfaces and different UNI clients were attached with multiple cross connects. LSP generation request was sent from one client to other (from either end) and all messages traces were monitored using InterEmulator protocol analyzer view and Ethereal.

**Results:** 70-90 milliseconds

**T21 – Test name and description:** Time taken in LSPs generation in a comparatively large emulated network

**Test configuration and deployment:**

A total of twenty emulated devices were configured with E-NNI and I-NNI interfaces in between them. These devices included more than five UNI clients. LSP generation request was sent from one client to other (from either end) and all messages traces were monitored using InterEmulator protocol analyzer view.

**Results:** 200-300 milliseconds
**T22 – Test name and description:** Time taken in LSPs generation in a comparatively large hybrid network

**Test configuration and deployment:**

A total of eighteen emulated devices with two real devices were configured with E-NNI and I-NNI interfaces forming different domains as a hybrid network. These devices included more than five UNI clients. LSP generation request was sent from one client to other (from either end) and all messages traces were monitored using InterEmulator protocol analyzer view and Ethereal.

**Results:** 280-350 milliseconds

**Summary:** LSP generation time depends on memory and processing power of cross connects facilitating this LSP. If number of cross connects increase more memory is needed. Emulated network LSP generation seems to be fast because it works within same application contextual space of InterEmulator. LSP generation from real to real cross connect is slower than LSP generation from one UNI client to another connected with then same network of two real cross connects.

### 6.2.3 Tests for scalability

Scalability testing was performed on already deployed emulated and hybrid networks. Effect of OSPF-TE initialization and LSP generation time was observed when new cross connects and UNI clients were added in the topologies. These results were achieved by using E-NNI and I-NNI interfaces.

**T23 – Test name and description:** LSP generation time taken for 10 cross connects added in T21 (emulated network)

**Test configuration and deployment:**

Ten more cross connects were added in an emulated network defined in test T21. These cross connects were added within different domains using I-NNI interfaces.

**Results:** 400-550 milliseconds

**T24 – Test name and description:** LSP generation time taken for 10 cross connects added in T22 (hybrid network)

**Test configuration and deployment:**
Ten more cross connects were added using I-NNI interfaces within domains in pre-deployed network of test T22.

**Results:** 500-600 milliseconds

**T25 – Test name and description:** LSP generation time taken for 10 more UNI clients added in T21 (emulated network)

**Test configuration and deployment:**

In this test 10 more UNI clients were added in the network defined in test T21. These clients were added to different cross connects in different domains and result was observed.

**Results:** 220-300

**T26 – Test name and description:** LSP generation time taken for 10 more UNI clients added in T22 (hybrid network)

**Test configuration and deployment:**

In this test 10 more UNI clients were added in the network defined in test T22. These clients were added to different cross connects in different domains and result was observed.

**Results:** 300-370

**T27 – Test name and description:** Effect of increased cross connect on network initialization and link state database synchronization time (OSPF scalability) with one domain and area.

**Test configuration and deployment:**

A large network of 60 emulated nodes was configured including UNI clients. All devices were part of one domain and default area. All cross connects were connected via I-NNI interfaces. Different UNI clients were connected with different cross connects.

**Results:** Domain border selection took 20-30 seconds and link state database synchronization time was 23-30 seconds. So total up time for OSPF was observed 40-60 seconds.

**T28 – Test name and description:** Effect of increased cross connect on link state database synchronization time (OSPF scalability) with multiple domains and areas.

**Test configuration and deployment:**
A large network of 60 emulated nodes was configured including UNI clients. Devices were distributed into domains using E-NNI interface. Each domain contained maximum 10 devices including UNI clients. All devices within a domain were connected via I-NNI interface.

**Results:** Link state database synchronization time was less than 20 seconds. So total up time for OSPF was reduced by a larger factor.

**Summary:** OSPF database size on each node increases as the number of nodes in the local domain increases. Network initialization slows down when more cross connects are added. Size of link state database on each cross connect decreases making it more efficient and corresponding size on area borders increasing making them more responsible for the local area. LSP generation time increases linearly if the number of cross are increased. It matters a little whether destination UNI client was in local large area or in some other area. What actually matter is the number of cross connects in between two UNI clients that need LSP path in between them.

### 6.2.4 Stress tests

These tests were performed in context of scalability to check maximum number of LSP allowed on a network and maximum LSP loads that can be applied to a real device.

**T29 – Test name and description:** Maximum number of allowed LSPs in a hybrid network

**Test configuration and deployment:**

Eighteen and more UNI clients were added to a hybrid network of three devices. Two emulated devices used were piano and piano2. Real device used was a78sa in IMK. All the emulated clients were connected to the emulated cross connects. 10Gb link between real and emulated cross connects was used. LSP was established from one client to another. Each LSP holding time (life) was kept 1 minute. Total 200 LSPs were tried from one client to another in one minute.

**Results:** 64 LSP with container VC4 were created successfully.

**T30 – Test name and description:** Maximum number of allowed LSPs on a real device in one minute

**Test configuration and deployment:**

Configuration was exactly the same as defined in T30. LSP was established from one client to another. Each LSP holding time (life) was kept to zero. It means LSP was deleted as soon it as was created. Total 200 LSPs were tried from one client to another in one minute.
Results: Maximum 120 LSP were possible in one minute from one emulated UNI client to another through real device.

6.3 Conclusion

All the tests performed for interoperability were successful. Most challenging part of the thesis was identifying and fixing interoperability problems between Alcatel and InterEmulator. After interoperability problems between Alcatel and InterEmulator were fixed all the tests gave positive results. Many tests in context of performance and scalability were performed. Network performance for initialization time and LSP generation time was calculated. Scalability of OSPF-TE link state database and LSP generation was studied. Almost all the tests were performed on emulated, real and hybrid networks. In the next chapter a conclusion of overall thesis and tests is presented.
7 Conclusion and outlook

This chapter provides the concluding remarks for the thesis. It is divided into two parts:

- Achievements
- Outlook

7.1 Achievements

This thesis was started with the aim to deploy and evaluated real/emulated optical network equipment available in VIOLA testbed. There were a lot of interoperability problems among different vendor devices. Special focus was on emulated devices interoperability with Alcatel 1678 cross connects. All the interoperability problems were solved successfully and a hybrid network was established for evaluation purposes. Successful evaluations in context interoperability, performance and scalability were performed on this deployed hybrid network.

Interoperability tests between Alcatel and Sycamore devices resulted into negative outcome. Both vendor devices could not communicate within the VIOLA environment and both vendors were prepared to work on interoperability. Interoperability between Alcatel real device and InterEmulator defined emulated devices were successful. Devices could communicate with or without IP tunnels. Different network scenarios deployment was possible. Typical network deployment scenarios were defined and deployed. These scenarios include real device connectivity with emulated devices. LSP generation worked for all kinds of hybrid networks including single domain network with I-NNI interface and multi-domain network with E-NNI and I-NNI interfaces.

Performance tests between emulated, hybrid and real networks resulted into interesting results. Network initialization and LSP generation times were observed for emulated, hybrid and real networks. Network initialization consisted of two major things: RSVP up time and OSPF up time. In OSPF up time different aspects like OSPF link state database synchronization time and size were important things for performance evaluation. Different scenarios of real and emulated devices connectivity were tested in context of performance. It was observed that adding cross connects decreases the network initialization performance. As a result of such increase in number of cross connects linear increase in RSVP and OSPF network initialization times happens. On the other hand, if UNI clients are added to a network it has nothing or a minimal effect on the performance. However, it affects the OSPF link state
databases size because TNA addresses for newly added clients are distributed over the network, so the database size increases as the number of UNI clients increase.

It was observed that keeping areas or domain small results in fast network initializations. If a network of fifty devices is constructed with forming one domain and one area then the link state database synchronization time is considerably larger than the time taken by the same event on the same network divided into small areas and domains. Database synchronization messages sent over the network floods the network. However, if the network is divided into small areas and area border routers are defined then network messages flooding don’t occur and OSPF database size reduces on each cross connect. Size of OSPF database increases on area border routers and summary messages (related to link and addresses information of all routers in local area) are sent between these area border routers. This phenomenon strengthens the idea deploying network into small areas and domains.

In case of LSP generation, time taken in establishment of LSP from one client to another is directly proportional to the number of cross connects. This time also increases linearly as the number of cross connects increase. However, increasing number of UNI clients over the same network does not affect much in this context. So increase in number of cross connects increases the LSP establishment time but increase in UNI clients does not increase this time.

An important observation in context of LSP generation was number of LSP generated per minute. Such results provide maximum threshold value for LSP establishment from InterEmulator to Alcatel real device. Up to two hundred LSPs per minute were tried from different emulated UNI clients defined in InterEmulator to Alcatel 1678 cross connect in IMK. Holding time for LSP for one minute gave maximum LSPs that were possible in a given situation. It was observed that maximum number of LSP generation depends on link bandwidth, chosen SDH container and number of available interfaces. However, when LSP holding time was reduced to zero (i.e. LSP is removed just after it is established) it was observed that maximum 120 LSP per minute could be generated from InterEmulator to Alcatel real device.

Furthermore it is worth mentioning that no model can be extracted in order to map readings taken from a completely emulated network to those taken from real or hybrid networks. This is because of the performance limitation of InterEmulator. However, general behaviour or approximated tendencies of the network can be observed from a completely emulated network.

Use of InterEmulator emulated environment was very successful and useful for interoperability study. Performance study using such an environment is beneficial for small
networks. InterEmulator, running on a single machine, is not suited for scalability study of large networks. A network deployed on single machine hosting InterEmulator needs more resources than are available on a single high resource machine. So a large network of emulated or real devices emulated on such InterEmulator environment becomes very slow and does not provide true results for performance and scalability. It was observed that deploying a network of sixty or more devices becomes slow. Normal network initialization and LSP generation takes more time than expectation derived from small network on the same InterEmulator environment. A truly large network of one to three hundred devices could not be established because of performance issue.

This performance problem is not only true for InterEmulator but it should be true for all emulated environments because for each emulated device a full stack of network protocols has to be loaded that needs memory and CPU cycles for each network request.

### 7.2 Outlook

True scalability study for large networks is not possible with network emulation environments like InterEmulator. This is because each network device requires its own processing and memory resources. If hundreds of devices are emulated on a Solaris box then it is obvious that such emulated network will not perform as efficiently as real devices network can.

InterEmulator environment can work as a distributed system. This system makes a distributed network of client server machines. Cross connects can be emulated as clients on cluster of machines and they all can communicate and report of a central InterEmulator server. One InterEmulator client machine can be used to emulate only one client or very few cross connects can be emulated on it. Network deployment in this way may provide a good emulated environment for scalability study.

Large topologies for real research networks e.g. X-WIN can be deployed in a well defined distributed scalability-friendly InterEmulator environment to check how this network behaves when GMPLS/ASON and UNI standards are utilized.
References

1. Aboul-Magd, B. Jamoussi, “Automatic Switched Optical Network (ASON) Architecture and Its Related protocols”, IETF Internet-Draft <draft-ietf-ipo-ason-02.txt>, 2002.

2. Adrian Farrel, Jean-Philippe Vasseur, Arthi Ayyangar, “A Framework for Inter-Domain MPLS Traffic Engineering”, IETF Internet-draft <draft-ietf-ccamp-inter-domain-framework-04.txt>, 2005.

3. Alcatel University, “1678 MMC GMRE Introduction”, Alcatel training manual, 2005.

4. Alcatel University, “1678 MMC GMRE User Guide”, Alcatel training manual, 2005.

5. B. Rajagopalan, J. Luciani, D. Awduche, “IP over Optical Networks: A Framework”, IETF RFC 3717, 2004.

6. Ben Schultz, “Understanding MPLS”, UNH Interoperability Lab slides, 2002.

7. Biswanath Mukherjee, Dhritamin Banerjee, S. Ramamurthy, and Amarnath Mukherjee, “Some Principles for Designing a Wide-Area WDM Optical Network”, IEEE/ACM Transactions on Networking VOL. 4 No.5, 1996.

8. Chuck Semeria, “RSVP Signalling Extensions for MPLS Traffic Engineering”, Juiper White Paper - www.juniper.net, 2000.

9. D. Awduche, L. Berger, D. Gan, T. Li, V. Srinivasan, and G. Swallow, “RSVP-TE: Extensions to RSVP for LSP Tunnels”, IETF RFC 3209, 2001.

10. D. Katz, K. Kompella, and D. Yeung, “Traffic Engineering (TE) Extensions to OSPF Version 2”, IETF RFC 3630, 2003.

11. D. O. Awduche, “MPLS and traffic engineering in IP networks”, IEEE Communications, 37(12), 1999.

12. Daisaku SHIMAZAKI, Eiji OKI, Kohei SHIOMOTO, and Naoaki YAMANAKA, “Scalable Multi-Layer GMPLS Networks Based on Hierarchical Cloud-Routers”, IEICE TRANS. COMMUN., VOL.E88–B, NO.3, 2005.

13. E. Mannie (ed.), “Generalized Multi-Protocol Label Switching (GMPLS) Architecture”, IETF RFC 3945, 2004.
14. E. Rosen, A. Viswanathan, R. Callon, “Multiprotocol Label Switching Architecture”, IETF RFC 3031, 2001.

15. International Engineering Consortium Tutorials, “Generalized Multi-protocol Label Switching (GMPLS)”, Web ProForum Tutorials - http://www.iec.org/online/tutorials/gmpls.

16. J. Drake, D. Papadimitriou, “Generalized MPLS (GMPLS) RSVP-TE Signalling in support of Automatically Switched Optical Network (ASON)”, IETF Internet-Draft <draft-ietf-ccamp-gmpls-rsvp-te-ason-04.txt>, 2005.

17. J. P. Lang, Y. Rekhter, D. Papadimitriou, “RSVP-TE Extensions in support of End-to-End Generalized Multi-Protocol Label Switching (GMPLS)-based Recovery”, IETF Internet-Draft <draft-ietf-ccamp-gmpls-recovery-e2e-signaling-03.txt>, 2005.

18. Jim D. Jones, “RSVP Extensions for User Network Interfaces (UNI) 1.0 Signalling, Release 2”, Optical Internetworking Forum, 2004.

19. K. Kompella, Y. Rekhter, “Signalling Unnumbered Links in Resource ReSerVation Protocol - Traffic Engineering (RSVP-TE)”, IETF RFC 3477, 2003.

20. K. Kompella, Y. Rekhter (eds.), “OSPF Extensions in Support of Generalized Multi-Protocol Label Switching,” IETF Internet-draft <draftietf-ccamp-ospf-gmpls-extensions-12.txt>, 2003.

21. L. Berger, Editor, “Generalized Multi-Protocol Label Switching (GMPLS) Signalling Functional Description”, IETF RFC 3471, 2003.

22. L. Berger (ed.), “Generalized Multi-Protocol Label Switching (GMPLS) Signalling Resource ReserVation Protocol-Traffic Engineering (RSVP-TE) Extensions”, IETF RFC 3473, 2003.

23. Lixia Zhang, Stephen Deering, Deborah Estrin, Scott Shenker and Daniel Zappala, “RSVP: A New Resource ReSerVation Protocol”, IEEE Networks, 1993.

24. Loa Andersson, Ina Minei, Bob Thomas (eds.), “LDP Specification”, IETF Internet-draft <draft-ietf-mpls-rfc3036bis-03.txt>, 2005.

25. Navtel, “InterEmulator 6.1 User Manual”, Navtel Communications Inc. Canada, www.navtelcom.com, 2005.

26. Nic Larkin, “ASON and GMPLS – The battle of the optical control plane”, Data Connection white paper - www.dataconnection.com, 2002.
27. OIF, “Creating an Intelligent Optical Network Worldwide Interoperability Demonstration”, OIF world Interoperability demonstration, 2004

28. P. Ashwood-Smith, L. Berger (eds.), “Generalized Multi-Protocol Label Switching (GMPLS) Signaling Constraint-based Routed Label Distribution Protocol (CR-LDP) Extensions”, IETF RFC 3472, 2003.

29. Paul Brittain, Adrian Farrel, “MPLS Traffic Engineering: A choice of signalling protocols”, Data Connection white paper - www.dataconnection.com, 2000.

30. Peter Kaufmann, Ferdinand Hommes, “VIOLA: A testbed for advanced network services”, VIOLA Internal document, 2004.

31. Telecommunication Standardization sector of ITU, “Architecture for the automatically switched optical network”, ITU-T document G.8080/Y.1304, 2001

32. VIOLA project tests, “OIF E-NNI Interworking Test specification – Alcatel/Siemens”, VIOLA Internal document, 2005.

33. Zafar Ali, Anca Zamfir, D. Papadimitriou, “Component Link Recording and Resource Control for GMPLS Link Bundles“, IETF Internet-draft <draft-zamfir-explicit-resource-control-bundle-05.txt>