ICT-216372

Trilogy

Trilogy: Re-Architecting the Internet.
An Hourglass Control Architecture for the Internet, Supporting Extremes of Commercial, Social and Technical Control

Large Scale Integrating Project

FP7 ICT Objective 1.1 – The Network of the Future

D3 – Initial Overall Architecture

Due date of deliverable:31st August 2008

Actual submission date: 31st August 2008

Start date of project: 1 January 2008 Duration: 36 months

Lead contractor for this deliverable: Roke Manor Research Ltd

Version 1.0, date 31st August 2008

Confidentiality status: Public

Abstract

This document presents an initial proposal for the architecture of the Future Internet. The architecture has been developed by the Trilogy collaborative research project, whose focus is the development of the current Internet, and in particular its network and transport layers. The key challenges identified include scaling and flexibility in the global routing system, and resilience and resource management to support the needs and demands of emerging applications. As well as the strictly technical aspect, the new architecture aims to incorporate the necessary flexibility to adapt to changing economic and social stresses, the so-called "design for tussle". While the architecture and Design Principles are presented here as an evolutionary step from the original Internet concepts, the implications for future network evolution are wide ranging. The architecture is currently in an initial form, and this document also highlights the open research issues and necessary validation activities, as well as the relationship to other ongoing Future Internet research.

Target audience

The primary target audience for this document is the networking research and development community, particularly those with an interest in the Future Internet. The material should be accessible to any reader with a background in packet switched network architectures. This document will also be of interest to those concerned with the interactions between network architectures and their economic, social and regulatory context, although specialist expertise in these areas is not a pre-requisite.

Disclaimer

Impressum

Full project title:	Trilogy: Re-Architecting the Internet. An Hourglass control Architecture for the Internet, Supporting Extremes of Commercial, Social and Technical Control
Document title:	D3 – Initial Overall Architecture
Editor:	Robert Hancock, Roke Manor Research Limited
Project Co-ordinator:	Mat Ford, BT
Technical Manager:	Phil Eardley, BT

This project is co-funded by the European Union through the ICT programme under FP7.

Copyright notice

Executive Summary

The Internet as we know it today is orders of magnitude larger and more complex than was ever considered during its initial design several decades ago. In large part, it owes this success to the clarity and simplicity of the original engineering model: more sophisticated network architectures have come and gone, and in the meantime the Internet has continued to grow and evolve, often in unexpected ways. However, we may soon reach a stage where continued growth and innovation are stifled by the constraints of currently deployed networks; indeed, some commentators say this stage has already been passed. Trilogy is a three-year collaborative programme to research new networking technologies to address these problems, and this report describes the initial architectural framework that has been developed to guide the work.

This Internet paralysis has not been caused by any fundamental misconceptions in its original design. Rather, it is that the role of the network and the demands placed on it have expanded in scope so significantly: instead of serving a single community unified by a common purpose, the Internet now spans the global economy, where numerous stakeholders naturally seek to exploit it for their own purposes. Rather than attempt to redesign the Internet for a particular business model, we seek to develop an architecture that can adapt to the stresses of economic and social change, whichever direction they take, an approach christened by the original Internet architects as 'design for tussle'. We also seek to extend the architecture to accommodate new concepts from the networking research community, to enable the Internet to meet new scaling challenges of network size and speed, and to support new classes of application.

We approach the problem of developing a new architecture from two directions. On one hand, we have revisited the original Internet Design Principles, and re-evaluated them from the perspective of the architectural problems visible today. Our result is a set of four refined Design Principles:

- Exposure of Information
- Separation of Policy and Mechanism
- Fuzzy Ends
- Resource Pooling

These should be seen as complementing and extending the original Internet Design Principles, and are a first stage of capturing the abstract concept of 'design for tussle' as a more concretely applicable engineering methodology. Our second avenue of investigation has been to take key novel networking techniques (especially applicable to the reachability and resource control problems), and analyse them for their architectural implications, especially points of compatibility and contradiction with the current Internet. These techniques include:

- the use of alternative global routing schemes with split identifier and locator spaces
- end-to-end transport protocols that can exploit multiple paths
- resource control mechanisms that naturally incorporate finer-grained response to congestion and accountability enforcement.

Taking these two approaches together, we hypothesise a baseline Trilogy architecture, which is comparable in scope to the current Internet network and transport layers, but with a subtly different internal structure. In particular, we divide the functions involving the networking infrastructure into two planes, for reachability and forwarding, and distinguish them from the transport services to which the network is totally transparent. In this decomposition, we have taken care to be clear about exactly which identifier spaces are fully visible to which components, and which are opaque or invisible. As well as describing this component structure in abstract terms, we have also outlined a mapping to the current Internet: we do not expect that any clean-slate architecture will be deployed in the real world,

but it can serve as a beacon for the evolution of existing networks, and the mapping highlights compatibilities and mismatches.

Development of an architecture is essentially an artistic process: it cannot be formally derived from requirements or principles, or proven to be the unique solution of an engineering problem. Continuous validation of the design is therefore a critical process. We approach that question again from two perspectives. Firstly, we look for compatibility at the engineering level between the architecture and particular examples of valuable network technologies. In that analysis, we find that although many novel techniques can be mapped directly into our connectionless baseline, there are several areas where further development is required: in particular, many approaches to the problems of routing scalability and denial of service protection suggest giving the network infrastructure some awareness of connection-like relationships between end systems. Finding a natural architectural paradigm to capture such functionality is ongoing work.

The second validation question is whether our architecture meets the goal of design for tussle. In that area, the most rigorous technique currently available to us is to see whether the design respects the lessons learned from current Internet evolution. We have previously reported a large set of case studies on network technology development from a tussle perspective, and a set of high-level design goals derived from them; this document builds the bridge between those goals and the Trilogy Design Principles, and in particular shows which social and economic requirements have to be directly reflected at the engineering level of the architecture, and which can be satisfied externally. In the future, we plan economic modelling to validate this more formally.

List of Authors

Alan Ford	RMR
Arnaud Jacquet	BT Group
Barbara van Schewick	SLS
Bob Briscoe	BT Group
Damon Wischik	UCL
Henrik Lundqvist	NEC
Iljitsch van Beijnum	UC3M
Ken Richardson	RMR
Lars Eggert	Nokia
Louise Burness	BT Group
Marcelo Bagnulo Braun	UC3M
Mark Handley	UCL
Matthew Ford	BT Group
Olaf Bonness	DTAG
Olivier Bonaventure	UCL-BE
Philip Eardley	BT Group
Pierre Francois	UCL-BE
Robert Hancock	RMR
Rolf Winter	NEC
Sébastien Barré	UCL-BE
Toby Moncaster	BT Group
Vamsi Kambhampati	Nokia

Table of Contents

List of Figures

List of Tables

Abbreviations

ADSL	Asymmetric Digital Subscriber Line
AQM	Active Queue Management
AS	Autonomous System
ATM	Asynchronous Transfer Mode
BGP	Border Gateway Protocol
CDMA	Code Division Multiple Access
CDN	Content Distribution Network
DCCP	Datagram Congestion Control Protocol
DDoS	Distributed Denial of Service
DPI	Deep Packet Inspection
DSLAM	Digital Subscriber Line Access Multiplexers
ECMP	Equal Cost Multipath
ECN	Explicit Congestion Notification
HIP	Host Identity Protocol
IDIPS	ISP-Driven Informed Path Selection
IGP	Interior Gateway Protocol
ISP	Internet Service Provider
LISP	Locator ID Separation Protocol
NAT	Network Address Translator
OSPF	Open Shortest Path First
P2P	Peer to Peer
PCN	Pre-Congestion Notification
PNNI	Private Network Node Interface
PSTN	Public Switched Telephone Network
PTR	Proxy Tunnel Router
RCP	Rate Control Protocol
RTT	Round Trip Time
STUN	Simple Traversal of UDP through NATs/Session Traversal Utilities for NAT
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
VPN	Virtual Private Network
VoIP	Voice over IP
XCP	Explicit Control Protocol

1 Introduction and Context

This document captures the initial results of the Trilogy project towards re-architecting the Internet. We present our initial overall architecture and stress that these are interim results produced half-way through the first year of a three-year project. This work will be updated in 12 months time.

Readers may be prompted by the title of this document to ask what we mean by architecture. We use the term 'architecture' to mean two things specifically. Firstly we mean our Design Principles, which are guidelines for Internet engineers and researchers. Secondly, we mean our holistic design for network and transport control. Others use the term 'architecture' to mean all sorts of things!

The purpose of the Design Principles is to provide simple guidance for Internet engineers, architects and researchers who are designing new Internet control mechanisms or considering modifying the Internet control architecture. Our aim is to update the original Internet Design Principles (such as the end to end principle) in the light of the new Internet environment, in particular so it is 'designed for tussle', whilst still fulfilling the original Internet design goals. As such it takes the high level goals, which were the results reported in [Trilogy08], and transmutes them into engineering level guidelines.

The purpose of the Control Architecture is to provide a unified control architecture, in other words an "hourglass for control". The scope is the traditional network and transport layers. The control architecture organises a set of baseline assumptions into a coherent technical picture and as such represents an initial application of our Design Principles. It also allows us to formulate hypotheses at the architectural level about how to solve known and emerging problems.

Both the above aspects provide a framework within which the future research work of the project can be carried out, and within which we can consider issues about our approach and questions about our specific protocol proposals.

The intended readership for this report is quite general – other Internet researchers, protocol designers, network architects and engineers. We will continue to seek their feedback through dissemination of specific aspects of the Design Principles and architecture at conferences and the IETF and so on. The architecture will change as a result of this feedback and also our own work within the project to validate the conclusions and develop specific technical solutions.

The objective of the Trilogy project is to develop new solutions for the control architecture of the Internet that remove the known and emerging technical deficiencies while avoiding prejudging commercial and social outcomes for the different players. The focus is the generic control functions of the Internet – the neck of the hour-glass, but for control.

1.1 Architectural Stresses in the Internet

While the Internet has radically changed business and society over the past decades, its architecture has hardly evolved. But now pressure from new applications, new business models and new networking technologies is distorting the architecture to such a degree that a redesign is now required.

We believe that two fundamental pieces of work need to be done. Taken together, their results will significantly enhance the reliability, robustness, manageability and functionality of the Internet, and will create new and varied business opportunities based around a common control architecture.

The first fundamental piece of work is how to develop *an architecture for change*, meaning an architecture that can adapt in a scalable, dynamic, autonomous and robust manner to local operational and business requirements. It needs to avoid prejudging commercial and social outcomes for the different players so that it can reflect the outcome of contentions amongst the Internet's stakeholders: it is 'designed for tussle', to use the phrase coined in [Clark05].

Our first results here are in Section 2, which presents our initial Design Principles. Design Principles are really guidelines to help a protocol designer or network designer achieve a solution with desirable

properties. For example, one of the original Internet Design Principles [Clark88] is the well known "end to end principle", which helps a design achieve the beneficial properties of resilience and evolvability. Our aim is to translate the aspirational goal of 'design for tussle' into more concrete Design Principles that are usable by protocol designers, network architects, and other members of our target audience. We originally thought that we would come up with several new Design Principles to replace some of the original ones. Instead we have found that, in the main, we have simply had to adapt the original ones so that they become tussle-enabled:

Original Design Principles * new environment = tussle-aware Design Principles.

The second fundamental piece of work is to develop *a unified architecture for control* that removes the known and emerging technical deficiencies. The scope is the network and transport layers and in particular the functions of reachability and resource control, together with socio-economic control capabilities. The target here is at a somewhat more detailed engineering level than the Design Principles.

The sorts of "technical deficiencies" we're concerned about are:

- Scaling: for example the routing system, which is the single most critical part of the Internet infrastructure, is facing significant scalability issues. Certainly some people believe that lack of scalability will 'go critical' within the next few years [Cho06], [Meyer07].
- Non-transparency: middleboxes like NATs, firewalls and DPI boxes break the original Internet assumption (or Design Principle) that the network provides a transparent communications pipe between end systems.
- Feature interaction: the canonical example is the vicious cycle between multihoming and traffic engineering. Multihoming was not architected into the routing protocols, nor into the congestion control mechanisms. Therefore, neither handles it properly. To bridge this gap requires the active efforts of network operators, using crude traffic engineering tools that often serve to exacerbate routing problems – which in turn drives the need for yet more multihoming and traffic engineering.
- New control requirements: examples here are the requirements (or assumptions) made by end hosts and their applications about the Internet's resilience, timeliness, non-trustworthiness and mobility, when compared to the original Internet (best effort communications between trusting, static hosts). Techniques include multihoming, IntServ, IPSec and Mobile IP.
- New business models: A key reason for the deployment of middleboxes is that they enable new business models like tiered bandwidth pricing and walled gardens, which go beyond the primitive flat-rate subscription and settlement-free peering of the original Internet control architecture.
- Competing stakeholders: there is no longer a clear delineation of roles between the users, who implicitly request resources, and the providers, who supply the resources and implicitly work out how to share resources between the users. Instead, there is something akin to an arms race between users trying to get more and more resources and providers trying to re-gain control of resource allocation. The issue is particularly potent in the context of peer-to-peer networks [P2PI08], [TANA08], [ALTO08].
- Concept rot[1]: all the above factors have combined to slowly degradate the original architecture, so that it is increasingly difficult to reason about and its protocols need more effort to maintain and are more likely to cause unexpected knock-on effects.

Our first results here are in Section 3 and currently are a set of complementary elements that we believe are steps towards a new unified Internet control architecture. Bearing in mind that the Internet architecture has lasted forty years and few real changes have been deployed (despite many years of research)[2], it should come as no surprise that we do not as yet have a complete proposal. The flip side

[1] By analogy with "software rot".

[2] For example the only change to the IP protocol itself in the last decade has been ECN (explicit congestion notification) – and although widely implemented it is in reality hardly used.

of it being very hard to change is that there is growing acceptance globally that change must happen, so the project is well-timed to impact the future Internet.

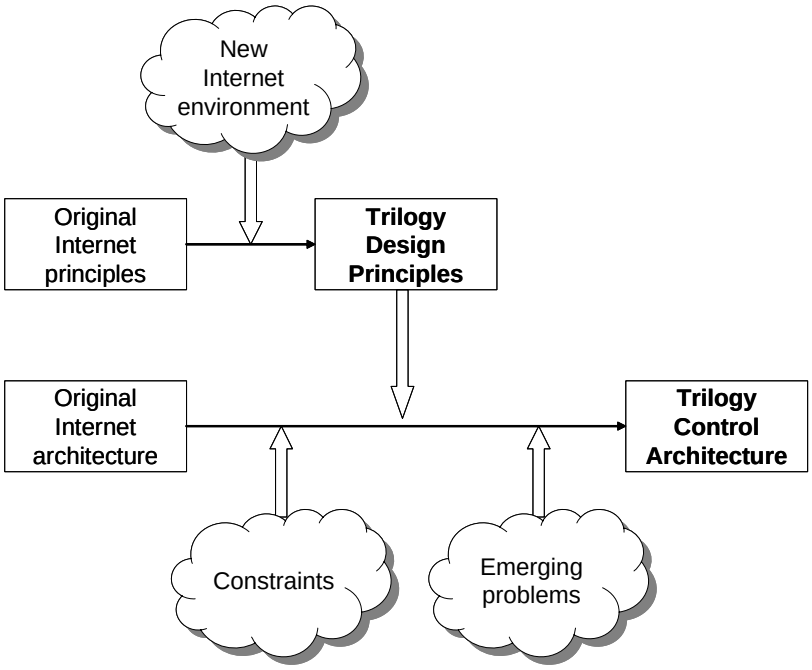

Figure 1: Towards a New Internet Architecture

1.2 Motivations for a New Architecture

Why are new Design Principles needed? Why is a new control architecture needed? Here we briefly describe the motivation for our work, and hopefully also motivate the reader to persevere to the meat of the document.

1.2.1 Design Principles

Originally the Internet was designed for a small number of cooperative, rational, expert end users who were all running similar applications (file transfer), attached to similar physical networks (low speed wired) and paid nothing. The Internet today is a very different world:

- Users have a huge variety of applications with very different bandwidth, latency and jitter requirements – from voice, web browsing, streaming, VPNs etc.
- There are far more users, with orders of magnitude more to come as ‘users’ become devices like sensor networks, i.e. without a human being in the control loop.
- Networks vary from kb/s access to Tb/s core; wireless and mobile networks are more and more common. Future access networks will increasingly be either optical or multi-standard software radios.
- Users may have no configuration freedom (company employees); yet even inexpert home users can download freeware code for system or applications.
- Users are self-interested (I want my stuff as soon as possible) and may even be malicious.
- Users may not even be rational or their rationality is quite different. For example, DDoS attackers and spammers may be anarchists or, more often, are trying to extort or trick money from people.
- Home consumers and corporates pay, whilst free networks like open WiFi are also common.
- There is a diversity of business models from walled gardens to community networks to bundled service providers.

The two key themes to the story above are firstly diversity in demand on the functionality of the Internet – diversity in all sorts of dimensions as described above – and diversity which will continue to increase. And secondly, the Internet is now an arena for conflicts among its stakeholders, such as users, ISPs, service providers and even regulators. In addition, on the Internet things change very fast in terms of: capabilities (e.g. processing and bandwidth – Moore's law), classes of applications and value chains (e.g. in the last five years, social networking, VoIP, e-commerce); software (instant deployment of new code); and so on. Therefore we believe that the future Internet must be designed to adapt to the local operational and business requirements at 'run time'[3]. In other words, the architecture must be 'designed for tussle' so as to allow local stakeholders (regulator, operator, users etc.) to adjust the local settings to meet local desires. This will result in different outcomes in different jurisdictions, in different markets, and for different users - to accommodate the fact that the control requirements will vary in different places and at different times.

However, 'design for tussle' is quite an abstract goal. In this report we try and translate it into more concrete, engineering Design Principles. Just as the goals of the original Internet design philosophy [Clark88] (interconnection of existing networks, survivability, support for multiple communications service types and physical networks) translated into several (inter-linked) Design Principles such as fate sharing, self-describing datagrams, soft state and the end to end principle. However, these goals have become increasingly less well met as the diversity described above has increased; nor is the 'design for tussle' goal being effectively and elegantly met today. So essentially our work is to update these Design Principles in the light of the new Internet environment described above, in particular so that following the new principles should lead to a design that is designed for tussle, as well as still fulfilling Clark's original design goals.

1.2.2 Control architecture

The basic Internet control architecture is the well-known hourglass shape depicted on the left in Figure 2. However, the hourglass picture of the Internet architecture is really about the data plane and omits the mechanisms needed for control. While people have easily been able to add new protocols and integrate new technologies into the data plane, it has become increasingly difficult to add control mechanisms to the architecture, leading to the "fat waist" shown on the right in Figure 2.

The three traditional cornerstones of Internet control are peering and policy configuration (on long timescales), routing path selection (on medium timescales) and congestion control (on short timescales). In a less heterogeneous world, with traditional applications, mostly wired network links, and relatively low expectations of service quality and reliability, these three were sufficient, and as they acted on different timescales, the interactions between them could be managed manually. This world no longer exists.

We believe that a future Internet architecture needs to jointly consider the reachability, resource control and socio-economic aspects, in order to simplify its control plane. The aim is to achieve an *hourglass for control*, and hence restore to the Internet its original elegant properties of transparency and evolvability, so that it once again copes with unforeseen growth and change. Whilst doing this, it needs to answer the list of problems raised earlier and also, by striving to follow our Design Principles, be designed for tussle.

[3] We refer to 'run time' loosely as the time after which the protocol or system is initially deployed.

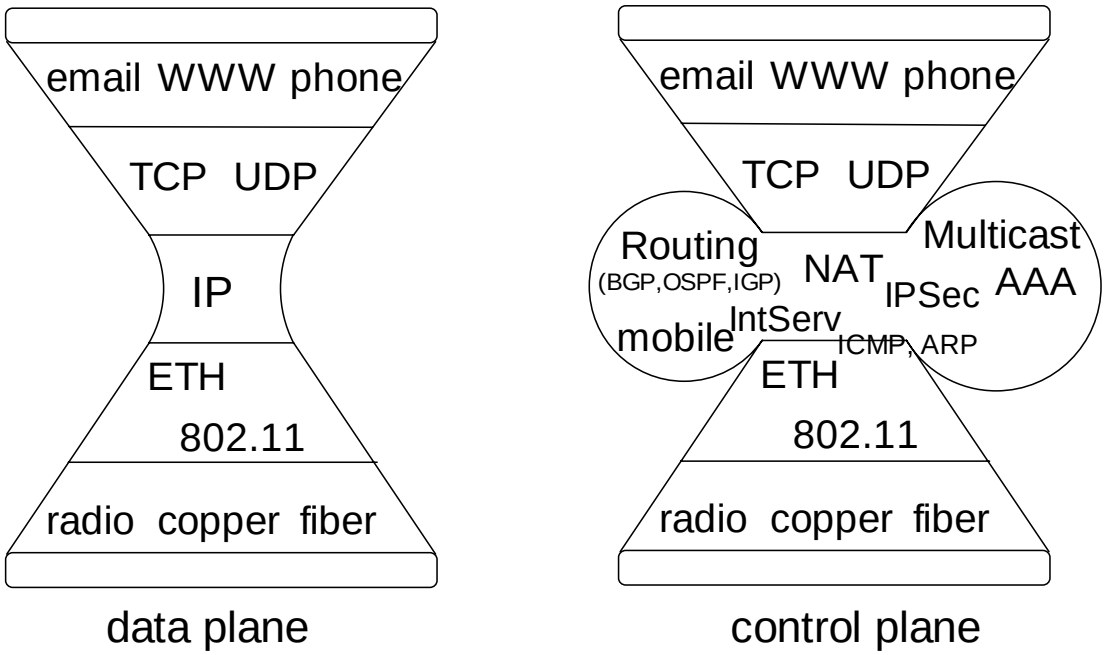

Figure 2: The Internet Data and Control Planes

1.2.3 Constraints on the new architecture

However, a new Internet architecture cannot simply be deployed by edict "The old architecture is dead, long live the new". We need to find a feasible path towards it, feasible both in that it is motivates those who have to change, and feasible in that it is technically plausible; this will be an open-heart surgery, and death of the patient during this operation is a non-optimal outcome!

In particular, we need to take account of the constraints that the current Internet sets: things that can't and shouldn't be changed. We have striven to do this in developing both our Design Principles and control architecture. Examples (we believe) are the concept of packets, and the idea that redundancy/diversity is the right technique to achieve robustness; whereas (we believe) it may be possible and right to change things like the default assumptions of implicit trust and end-to-end transparency.

1.3 Goals and Structure of this Document

Section 2 presents our Design Principles and Section 3 our Initial Architecture. The rest of the document builds on Sections 2 and 3. Section 4 explains how these results fulfil our high level objectives, especially the socio-economic ones, whilst Section 5 provides some first test cases to illustrate how to apply the Design Principles to some specific technical topics.

It is worth stressing again that this is our initial architecture (with an updated report in a year's time). The Design Principles will change as we gain experience of applying them within the project and of course discussing them with people outside Trilogy, be they researchers, standards people, business strategists, public policy experts, and so on. Similarly, the network/transport control architecture will build towards a more unified picture.

2 Design Principles for the Trilogy Architecture

This Section presents four engineering-level Design Principles. We outline them, explain what they mean and briefly look at how they relate to the original Internet engineering-level Design Principles of [Clark88]. This comparison is expanded in Section 4, together with a justification for the Design Principles in terms of how they enable our socio-economic aims – how they are 'designed for tussle'.

2.1 Information Exposure

The data (or transaction) that uses up scarce networking resources, through being sent (or acted on) should integrate control information (e.g. a metric) that reflects its resource usage in 'real time'. The control information should provide sufficient information about resource usage to support an efficient and accountable allocation of resources. The control information may be used as a variable in a pricing structure.

Some examples of the use of scarce networking resources are the forwarding of packets by a router ('data') and the creation /maintenance of a connection by a middlebox ('transaction'). The particular concern is those (economically significant) networking resources that are scarce, so their consumption by one user's data (or transaction) prevents another user consuming the resource; it is this externality (cost) that needs to be captured by the 'control information'. For example, where the scarce resource is a router's bandwidth, the control information is about the router's congestion. Other examples could be the shortage of table space for storing middlebox mappings or processing power for handling router messages.

Of course usage of resources which aren't scarce will (probably) need to be accounted for, but this information doesn't need to be "integrated" (see below) with the data /transaction. Also, resources which are purely used and optimised within a single system aren't relevant[4], e.g. resources purely at the end system (for the data /transaction).

'Integrate' means that we believe this control information (e.g. a metric) must be within the data (transaction) and not some kind of slow management message, i.e. it automatically synchronises the information with the resource usage:

- In time (over what timescale)
- In space (over which participants)
- In function (for what activity)

'Reflects its resource usage in real time' emphasises the first of these points, i.e. the granularity of the information in time should reflect that over which the data /transaction depletes the resources (in other words, the timescale it imposes load for). For a data packet, the time is the period for the router's queue to drain (typically considered to be about 1 RTT). For a transaction, if the scarce resource is storage of state, then it is the lifetime of the binding state.

For some types of resource, it is simply the data packet or transaction that consumes resource, regardless of its size. For others, the size of the data packet or transaction matters. For a data packet, the latter is typical (routers are 'byte congestible').

Practical considerations may spread the control information over time, e.g. ECN only allows one bit per packet and hence many packets must be monitored to get an accurate metric. One implication is that each node which uses up resource has to adjust the control information.

[4] There are some marginal cases where it is unclear if it is a networking resource consumed by others or a purely local node optimisation, for example in a wireless mesh a burst of traffic that causes some device in power saving mode to have to wake up to deal with the burst.

The second sentence in the Design Principle gives the reason for the control information ('support an efficient and accountable allocation of resources') and thus guides how much detail the information needs to have.

The third sentence reflects that market mechanisms are normally the right tool to mediate the allocation of resources amongst those competing to use them, and so are an effective way of achieving the goals of an effective and accountable allocation of resources. So we should consider carefully where the extreme case of "perfect competition" would drive us. The actual pricing formula may be off-line and may be sophisticated.

Our Information Exposure Design Principle can be seen as extending the original Internet Design Principle of connectionless datagrams (signalling goes in-band i.e. along with the packet) in two ways, in order to take account of the abstract goal of Design for Tussle. Firstly, by including the idea of accountability for usage of scarce resources, and secondly by recognising that transactions as well as packets consume resources.

Some open questions with this Design Principle

For a transaction, it may not be immediately obvious how much resource the transaction will consume, e.g. how much processing will it take to process the message? At one level, uncertainties are risks, which can also be handled as a cost.

Adding the integrated control information in itself consumes some resource. It is assumed that sufficient benefit is gained, for instance in terms of the efficiency with which resources are used.

2.2 Separation of Policy and Mechanism

Allow a network entity local choice according to its priorities (policy). Have a common protocol or method through which the policies interact to determine how networking resources are shared. Constrain conflicting policies via the Information Exposure Design Principle.

The policies are the implementation freedom that doesn't need to be standardised. This is both simple (the network entity knows what matters most to it, what its business models are, etc) and adaptable (adapt policy in light of its experiences). Implicitly, policies are independent of each other. On the other hand the mechanism does need to be standardised.

A good example is Kelly's view of congestion control [Kelly00]. The mechanism is congestion pricing, which represent the externality cost that a packet has on other users. It is the end user's choice how they react to this information, e.g. they prioritise some flows (which continue unabated) over others (which adapt their traffic rate down), or they continue as before (and pay more, in some direct or indirect fashion). It is also the policy decision of a network router when to congestion mark a packet (e.g. triggered by what level of traffic), as only it understands when its resources are starting to be depleted. Note that a congestion control approach where the network told the user what rate to run at (such as RCP [Dukkipati05]) would break this Design Principle.

Congestion marks also offer the opportunity for the network to create an admission control service, i.e. it uses the congestion marks to decide whether to admit or block a new flow admission request, in order to preserve the QoS of previously admitted flows. The network is free to set its own policies about how to translate the marks (the standardised mechanism) into its flow admission decisions [Eardley08]. Another example is DCCP [Kohler06]. It defines a three way handshake for connection set-up (mechanism), which allows the ends to negotiate which congestion control algorithm (TCP, TCP friendly etc) to use for the data transfer phase. The choice is independent for each pair of endpoints. Yet another example is BGP routing. The "mechanism" is that an Autonomous System offers paths to IP prefixes to its neighbouring ASs. The "policy" is that each AS selects (from the advertised ASs) its "best" path towards a destination prefix, based on its own criteria.

"Constrain conflicting policies": However, policies need to be sensible – for example, the congestion control policy set by one user can't ruin the service received by other users. We believe that enough

constraint is set by the Information Exposure Principle, i.e. the mechanism needs to integrate the control information (e.g. metric) that reflects the usage of scarce resource by the mechanism's data (or transaction). For example, policy about what granularity prefixes should be advertised /accepted (e.g. /16s or /32s?) – too fine a granularity causes too high a cost in terms of amount of messages and processing. In other words the policies are constrained to be sane by accountability.

Another point is that one shouldn't expect that two different mechanisms will give the same answer. For example, we no longer try to synchronise the OSPF topology view with the BGP topology view in a single-layered routing architecture - we have two layers in the routing architecture, and have separate mechanisms in each.

Our Modularity Design Principle can be seen as analogous to the original Internet Design Principle of 'IP over everything and everything over IP'. That defined a simple IP layer to transfer data, with freedom about the applications above and the networking technologies below. Here we're concerned about a similar modularity principle but for control – a common mechanism, with freedom over how the control information the mechanism provides is used ('above') and how the control information is set ('below').

Some open questions with this Design Principle

So far, detailed considerations have focussed on resource-related examples of the Design Principle. We need to work out how to capture reachability-related examples more formally.

Does the Information Exposure principle really give sufficient constraint on policies so that they don't conflict with each other? Detailed case studies are needed, including reachability related ones.

2.3 The Fuzzy End to End Principle

Allow the endpoint to explicitly delegate some functions into the network, so the end is effectively a distributed system. This may imply some state in the network, which should be "soft and hinty".

First, some examples of the sort of delegation we're referring to; these are things that all could be done by the endpoint, but that the network could perform as a 'useful service' for the endpoint:

- Application prioritisation: e.g. a DPI box estimates traffic priorities, in order that the customer makes best use of the capacity under their contract.
- Protection: a firewall filters out certain types of traffic, e.g. adult content, file sharing, games, chat rooms, unauthorised access etc. This could be for consumers (parental control) or businesses (control employee usage of Internet). Another example could be to restrict incoming access to authorised people only.
- Content caching and targeting: the network caches content from a server to speed up its delivery or to help the content supplier optimise the content for a particular user.
- Finding an appropriate peer: a network oracle may be useful in a peer-to-peer system, as suggested by [ALTO08], [P4P].

The "endpoint" might, for client server communications, be either end.

"Explicitly delegate" means that the endpoint has clearly delegated the function into the network, rather than the network imposing itself unannounced into the communications. Delegation is for a specific function or application. Note that "explicit" may or may not mean 'voluntary'. For example, a corporate network might insist that some functions are delegated from the endpoint to network nodes (middleboxes). Another example might be a broadband provider who insists some functions are delegated to it, as part of a service package[5]. For trust and accountability reasons it seems most likely that the delegation will be over a single administrative hop.

[5] Claims about equity or the usefulness of the service can be dealt with by competition and/or regulation.

"Endpoint" and "network" denote a functional relationship between computers that 'use' the network and computers that 'provide' the service offered by the network (i.e. data forwarding). So end points may well be administered by a network provider or topologically in the network.

One issue for the Design Principle is the constraints there are on the kind of function that can be delegated. One view is that delegation should only be allowed to nodes acting as application-level intermediaries, which "are an integrated part of the application, correctly terminate the protocol stack and implement their functionality at layers above the Internet layer" [vanSchewick09]. An alternative view is that delegation doesn't just apply to application-level functions, for instance most middlebox functionality would fall into this category. The latter case leads to some loss of transparency of the network (it is no longer purely a packet delivery mechanism), and some loss of flexibility for the network and the endpoints (and their applications). To lessen this, we recommend that the network state is "soft and hinty" – in particular, that loss of the state should not stop the end system from being able to communicate.

The principle seeks a compromise between network providers' interests to increase their profits by offering additional services to users, and the recognition that implementing functions in the network can affect the transparency of end to end communications. The Internet's stakeholders will tussle for economic and social control, but we want to try and stop an arms race between users and networks. We seek a more cooperative model where endpoints delegate and the network helps. We also want to try and make sure that the system can evolve by drawing these functions back into the endpoints. The "distributed system" view attempts to achieve the above points.

Our fuzzy end to end Principle is, unsurprisingly, closely related to the original Internet end to end Design Principle. As [vanSchewick09] points out, the end to end principle in fact comes in two versions. The "narrow" version of 1984, which says a function should not be implemented in a lower layer (i.e. in the middle) if it cannot be completely and correctly implemented at that layer, except as a performance enhancement. The "broad" version of 1998, which says lower layers (the middle) should only provide very general services that can be used by all applications; optimising support for one application hinders long-term system evolvability, application autonomy and reliability. The view that this Design Principle is restricted to application-level intermediaries is compliant with the "broad" version. The alternative view isn't restricted like this. In terms of the "narrow" version of the end to end principle it argues that functions can be implemented in the middle also for business /social reasons; and it modifies the "broad" version in a confined, controlled manner so as to minimise loss of its benefits.

Some open questions with this Design Principle

The main open issue has already been mentioned, i.e. whether the Design Principle applies only to delegation to an application-level intermediary, or also to other functions.

Another open issue is how to extend this Design Principle to something about more general delegation or relocation of control functions (a possible example might be BGP communities). This is a known desire, see [Trilogy08] but we don't yet have a theory about the set of techniques to achieve it.

2.4 Resource Pooling

Allow sufficient pooling of resources to be effective. Ensure that the resource pooling mechanisms don't conflict with each other.

Resource pooling means making the network's resources behave like a single pooled resource, i.e. separate resources appear to act as one large resource. So it implies the load (or a sufficient part of the load) can be "relocated" to a different resource or "spread" over several resources. "Relocation" and "spreading" can be in space or time, for example:

- In the analogue PSTN a phone call could set up a circuit using any of the empty channels on an analogue switch

- A packet switch makes available its entire capacity (not divided up into circuits), or at a later time, via buffering.
- Multihoming, multipath routing and traffic engineering pool together separate links (or networks) e.g. [Laws92]
- Google (and other CDNs) pools together the processing cycles, bandwidth and reliability of all its distributed servers
- A wireless system like CDMA pools the bandwidth, power and interference of its base stations and mobile terminals
- Network coding pools multiple messages into fewer messages over a pool of links [Katti06].
- With swarming downloads (e.g. BitTorrent) each receiver pools multiple other peers receiving the same data to act as if they are one data source; and it pools all the network paths from these peers.

The benefits of resource pooling are:

- increased robustness against component failures;
- better ability to handle localized surges in traffic;
- maximized utilization.

"Allow sufficient pooling of resources to be effective" means there is enough resource pooling to bring about the benefits. So there doesn't have to be unlimited relocatability. The "power of two choices" [Mitzenmacher01] suggests that in many cases it is enough if the choice is from a small set. Nor does it require that everything needs to be given a choice; sometimes it is enough if just a small fraction has choice. It may even be enough if the "spreading" doesn't explicitly involve a choice, for example with ECMP a specific flow follows a deterministic path, but overall it achieves reasonable load balancing.

"Conflict" (as in: "Ensure that the resource pooling mechanisms don't conflict with each other") means that two resource pooling mechanisms may work against each other, because they optimise for different criteria. For example ISPs shift their traffic to minimise their peering costs, but peer-to-peer applications want to download from the peer reached over the best performing path; it can be shown that the cost of anarchy (i.e. the degradation in performance due to conflicting load-shifting) can be arbitrarily high [Roughgarden02], [Acemoglu07]. A "conflict" might also be between two instances of the same resource pooling mechanism fighting to gain a bigger share of the pooled resource. It is assumed that conflicts can be handled via careful design (case by case analysis) and via the Information Exposure Principle.

Our Resource pooling Design Principle can be seen as extending and formalising the original Internet Design Principle that resilience[6] should be achieved through redundancy and diversity rather than through super-reliable individual components.

Some open questions with this Design Principle

What abilities does the network need to have in order to achieve effective pooling of a particular resource? For example, re-bind mappings so packets can be re-routed.

How to ensure that one resource pooling mechanism doesn't conflict with another? Some worked examples are needed; we have an idea how to resolve the specific conflict mentioned above [Handley08].

[6] Resilience is perhaps the main benefit of resource pooling. There are others such as being better able to handle a broad set of traffic matrices.

3 Initial Architecture

This Section presents the initial draft of the Trilogy architecture itself: it is within this framework that the future research work of the project will be carried out. We stress that the architecture given here is not complete but an outline, and in its outline form it is neither final nor definitive.

Rather, its first role is to organise a set of baseline assumptions into a coherent technical picture. The result, considering only its most fundamental properties, is not radically different from the classic Internet architectural vision [Clark88], [Carpenter96], although we give greater emphasis to clarifying some of the key functional boundaries. However, we recognise that the problems described in Section 1 pose hard challenges to the traditional Internet architecture, in particular in scalability and the accommodation of economic and social conflicts between the various stakeholders.

The second role of this initial outline is therefore to allow us to formulate hypotheses at the architectural level about how these problems can be solved. Our guiding principle is that the success of the Internet depends primarily on its underlying simplicity, and the broad applicability that directly follows. We therefore try to hypothesize the minimal extensions that can achieve the underlying project goals, rather than seeking generality as an architectural goal in itself.

The presentation of the architecture follows the following pattern. Section 3.1 defines the fundamental building blocks, and their justifications in terms of tussle boundaries. Section 3.2 describes a minimal connectionless architecture built from these blocks, and relates it to the current Internet, and Section 3.3 explains how more complex networking techniques can be incorporated in this same structure. Section 3.4 outlines a set of extensions, targeting directly some of the scaling and tussle problems of Section 1, which sacrifice some of the conceptual simplicity of the baseline; one of the main goals of the project is to evaluate these benefits and costs, and search for solutions with cleaner architectural properties. Finally, Section 3.5 considers the extent to which the initial architecture is successful in integrating the Internet's resource control and reachability functions.

3.1 Fundamental Building Blocks

A fundamental assumption of the architecture is that it is based on a minimal packet delivery service. The ideal case is that packets are entirely self-describing – that is, that other concepts such as connections, flows or sessions are higher level constructs, invisible to the delivery service, and the network delivers each packet independently of every other.

3.1.1 The Reachability Plane

We decompose the packet delivery functionality into two parts, the first of which is responsible for hop-by-hop outgoing link selection, and hence enables network-wide reachability. We call this part the reachability plane.

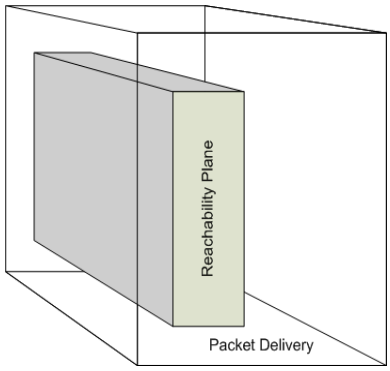

Figure 3: The Reachability Plane

The key identifier space for this component is the information needed to route each packet, which we take initially as a destination address or *locator*. Every allocated locator is reachable by default, and functions such as (D)DoS protection must be implemented at the receiver end system.

Are source locators required? While the source locator is often included in packet switching architectures, is it fundamentally needed in each packet? It has value in error reporting to the source (minimally for destination unreachability errors), and accountability enforcement (e.g. via ingress filtering), and local area networking schemes often use it for route table building. [Crowcroft08] discusses alternative architectures where the source address is not however required.

What other packet information influences the path? We want to isolate the source and destination nodes on one hand from the network infrastructure on the other, so the path itself is not visible in the packet format. However, path consistency between packets is an important property for endpoint-managed resource control algorithms (see Section 3.1.3), so we allow that other identifiers visible at the packet level may influence routing explicitly. In particular, we distinguish the concepts of *path selector* bits (further discussed in Section 3.3.1) and *service class* identifiers (also discussed in Section 3.1.3) for this purpose.

Are locators from a single global namespace? This approach has the greatest engineering simplicity, and indeed is the only approach totally compatible with the packets being fully self-describing. However, it is also the root cause of many of the Internet stresses, in that it immediately places all network participants (at least, packet sources and destinations if not forwarding infrastructure) into a single "tussle space". We consider possible relaxations in Section 3.3.3.

3.1.2 Implementation of the Reachability Plane

Apart from the packet source and destination, we assume that the reachability service is implemented by a set of autonomous, interconnected administrations or *domains*. The internal structure of each domain may be non-trivial, but is externally invisible: all that is exposed is information about which locator spaces are reachable and under what circumstances.

What reachability information or control is shared, beyond the locator spaces themselves? Current interdomain routing architectures include a diverse range of approaches to the exposure of information and sharing of reachability control between networks. On the one hand, basic BGP operations expose the global topology but fully hide policies of individual networks; conversely, community attributes can be used to relocate policy controls bilaterally between peers. We do not have a firm picture of what is required fundamentally. It is notable that much of BGP operation is concerned with traffic engineering (for load balancing), which is an area where the project is exploring solutions in a different part of the architecture (see 3.3.1).

What additional identifier spaces are used to manage the global topology, and how? We seek to minimise the use of locators in describing topology, to avoid extending that tussle space into inter-network operations. Inter-domain topology should ideally be described in terms of different identifiers, such as the Autonomous System (AS) number of BGP. Note that in interior routing protocols, router identifiers are typically IP addresses, but there is no actual need to couple these identifiers to the locators in the traffic being routed.

3.1.3 The Forwarding Plane

Alongside the reachability plane, network infrastructure needs to make decisions about how the transmission resource on each link will be apportioned between packets. We call this function the forwarding plane[7].

[7] Traditionally, this might be referred to as resource control. We use the term forwarding plane to acknowledge the fact that there may be other types of (non-transmission) resources; however, these are not managed inside the forwarding plane itself.

The fundamental packet delivery service is best-efforts: the network delivers packets to their destination, without guarantees on ordering or throughput. Nodes along the delivery path mark property information about the path into packets, either explicitly using bits in the header, or implicitly through delay or loss; end systems measure path properties and modify their sending rates in response. The allowed set of responses is left very open at this stage, and specific issues of accountability are discussed further in Section 3.3.4.

There is an implicit requirement that an end system generating a sequence of packets must be able to depend on consistency for their paths, at least over a scale of a round trip time or more. Reachability information is exposed to the forwarding plane only at this level of *path consistency*; in particular, detailed knowledge of locator syntax and semantics is not required – locators are opaque tags which only need to be tested for equality.

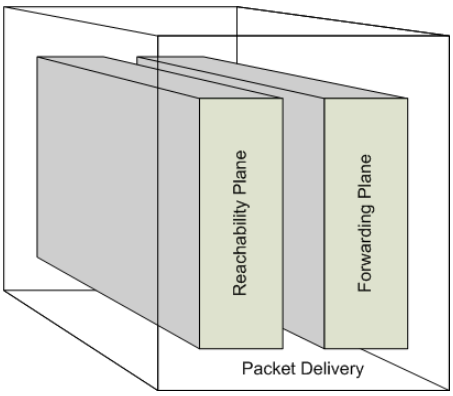

Figure 4: The Forwarding Plane

Looking at the information managed by the forwarding plane, we identify two major open issues.

What is the level of path property information provided by the network? The optimum situation, consistent with the principle of the reachability plane (that each packet is self-describing), is that each packet is marked with a complete description of forwarding resources available on the path, but this could impose a significant per-packet overhead (in size and forwarding cost). More constrained encodings impose stronger requirements for path consistency and stability for resource control loops to be effective.

How does the forwarding plane distinguish traffic types? We do, at a minimum, assume that any domain can define certain *service classes* with different forwarding performance, and that end systems select between these by embedding information in their packets. However, it is not clear if these service classes should or can be globally defined, or whether they are agreed only at interconnection points – between end systems and networks, and between networks themselves.

3.1.4 The Transport Services

Distinct from the packet delivery service, we identify the functions that are implemented in a pure end-to-end fashion. These define the communications service offered to applications (email, messaging, file sharing, multimedia, …). These functions, such as reliability, flow control or message framing, are totally invisible to the packet delivery service, and the identifier spaces involved are totally separate also. We assume the existence of *endpoint identifiers* which label the communicating parties[8], and that these are independent of the locators used in the reachability plane. However, we do

[8] These communication parties are similar to the *entity* concept defined in FARA [Clark03b]. There is no fundamental assumption that they are tied to particular types of node.

not require that the endpoint identifier space is global or coordinated. Indeed, a single node might use several different identifier families, and they may be implicit rather than explicit.

Transport services are not the primary focus of the Trilogy project. Nevertheless, we assume that they are the route by which the packet delivery functions are actually exercised. Re-engineering of the standard transport services is therefore a main method to exploit the functions of the Trilogy architecture.

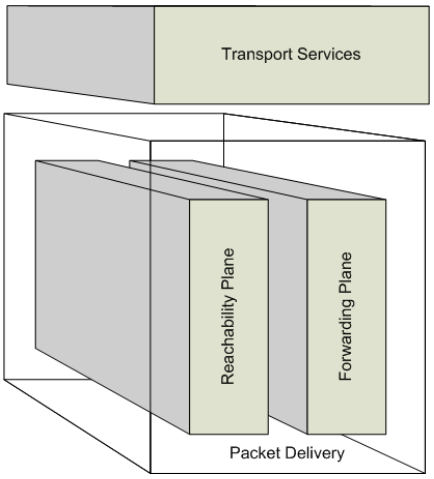

Figure 5: The Transport Services

3.2 A Connectionless Baseline

So far, the presentation has attempted to be agnostic about the location and packaging of the various functions into physical nodes, and this is a property that we intend to preserve. Many mappings are possible; Figure 6 below shows one illustration, where the functions are apportioned between two different types of node, the host and the router. The result aligns with many traditional presentations of the Internet architecture, and can be made still more prescriptive with the constraint that the network infrastructure consists entirely of routers, with hosts attaching only at the edges[9].

In this very simplified case, the functions can be apportioned between the two node types as follows:

- Applications, and hence the transport layer services, are contained entirely within hosts.
- The packet delivery services are distributed between both hosts and routers, but with a different subset in each:
    - The reachability plane in hosts is vestigial: the main requirement is to be able to determine the destination locator, and all packets can then be sent to the default router. Conversely, in routers the reachability plane encompasses the whole of routing protocol operation.
    - The forwarding plane in hosts is responsible for the rate control loop, including path property measurement. Conversely, in routers the functionality is the indication of resource loading information in the forwarded packets.

[9] We note that the Internet architecture itself does not absolutely require such a host/router distinction, or the assumed partitioning between the network infrastructure of operators, and the end systems of users. However, there is de facto support for it, e.g. in the existence of the host and router requirements RFCs ([Braden89], [Baker95]).

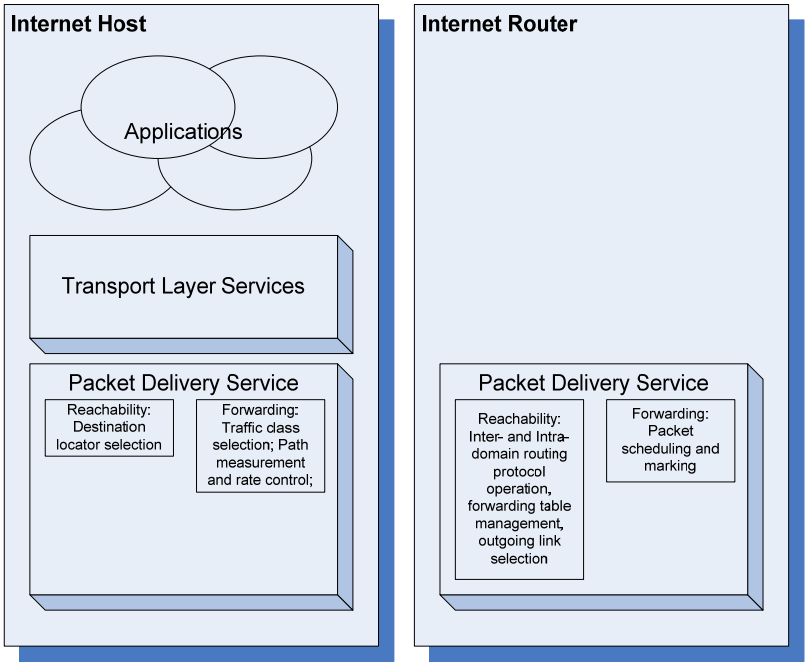

Figure 6: A Host/Router Architecture

A more detailed comparison of our fundamental architectural building blocks with today's Internet is given in the following table.

Building block	Concept or component	Internet Equivalent
Reachability Plane	Self-describing packets	IP (v4 or v6) datagrams; but note that middleboxes prevent these being fully self-describing.
	Locators	IP addresses. Both source and destination addresses are used in every packet.
	Single global namespace	There is a single managed IP address space, but unmanaged/private addresses are also used with interconnection through NATs.
	Path selectors	Not explicitly standardised, although ECMP (see e.g. [Thaler00]) considers hashing on a variety of header fields originally defined for other purposes. The flow-label field is a specific candidate in IPv6.
	Reachability	Nodes are assumed to be reachable by default. If middleboxes have to be specifically configured to allow packets, this has to be managed out-of-band.
	Autonomous infrastructure domains	Autonomous systems of BGP.
Forwarding Plane	Packet delivery	By default, a best efforts packet delivery service; possibly, some kind of service level agreement from the first-hop network provider.
	Multiple service classes	DiffServ allows several service classes to be defined and implemented locally (i.e. but within a single network), but the standardised service classes are only loosely defined and cannot be relied on end-to-end. IntServ signalling can create 'per-flow' service classes end-to-end (where supported), where the packet marking corresponds to fixing a set of header fields.[10]
	Load signalling	Drop, delay and packet marking can all be used, although the most common are packet drop and ECN (single-bit) marking.
	Resource control loops	For most traffic (TCP), there is an outer resource control loop which manages the size of a congestion window within the transport layer.
Transport Services	End-to-end reliability, flow control, …	Implemented separately (as required) in each transport layer; shares sequence number space and window control with the outer resource control loop.
	Separation from packet delivery service identifier spaces	The locator space often appears as part of the endpoint identity.
	Endpoint identifiers	Not explicitly defined and many different ones can be used (DNS names, HIP identities, home locators (Mobile IP).)

Table 1 : Comparison with the Internet

[10] The same basic technique would be possible in our minimal architecture; we would regard this as a user-initiated network management operation, and the resource control loop would still operate on the corresponding traffic (although it would only be effective in error cases).

3.3 Architectural Refinements

The architecture as presented so far has very limited sophistication, certainly compared to the complexity visible in current Internet operations. This is consistent with our primary goal of preserving simplicity at the architectural level, and formalising the degree of independence of the three main building blocks. Nevertheless, the same structure can accommodate a number of much more sophisticated networking concepts without affecting the underlying assumptions and boundaries; these concepts refine and clarify particular aspects of the architecture, but we do not regard them as extensions.

3.3.1 Resource Control Algorithms

The forwarding plane is responsible for controlling the rate at which packets enter the network from applications: conceptually, this functionality is extremely complex, and potentially binds all the network participants into a single tussle space of users seeking to maximise their throughput and providers seeking to manage their link utilisation. Decomposing this tussle space is a high priority. In addition to this social and economic challenge, there are increasing technical issues with the variety of different resource types that contribute to end-to-end packet transfer and whose allocation has to be managed, and the scaling of the corresponding algorithms to ever higher flow rates and link capacities.

Our fundamental technique for decomposing this problem space is the formalism developed in [Kelly98]. Within the forwarding plane, there are three sub-functions:

- Packet sources modify their transmission behaviour, so as to maximise a utility function U, subject to information p[11] about the path, reported back from the packet sink they are talking to.
- Forwarders write load information into each packet, so that this information integrates for each link as the packet traverses the delivery path.
- Packet sinks extract this information and report it back to the sender.

The critical points here are that the utility functions U are private to each user (and so do not need to be coordinated between themselves and the network), and that the details of the calculation of p are private to each forwarder (link). Only the syntax of p and how it is integrated along the path need to be globally standardised; the exact semantics of the sort of load it describes are only loosely constrained. The calculation of p and the definition of U are left to the local policies of each participant in the forwarding plane, and these policies can be essentially decoupled – the only requirement is that the utility maximisation respects the reported load. We use the existence of this technique to justify a more fundamental architectural assumption about the forwarding plane: ***Rate control by sources is governed only by local policies and network-derived load information signalled along the delivery path***. The remaining questions about resource control reduce to the following.

Firstly, there is the question about how to support a wider range of application types. The Internet today is dominated by best-efforts traffic for applications which are elastic, that is (following [Shenker95]) applications whose utility function U is solely a function of throughput. However, an overall project goal is to support a wider range of applications on the Internet, including inelastic traffic (requiring hard throughput guarantees), and what might be termed semi-elastic traffic (where the utility function depends on other performance parameters, such as delay). We have already noted above (Table 1) how to incorporate resource reservation on the IntServ model into this framework, which supports the inelastic case; note that we assume the packet marking and feedback control loop continue to operate even on this traffic, but they come into effect only in the case of errors (link failures and/or network rerouting). For semi-elastic traffic the state of the art is TCP-friendly rate

[11] In the standard presentation, this information p is referred to as a *congestion price*. We try to avoid the term "price" to eliminate confusion with real monetary prices, which are also in the scope of the project.

control ([Floyd00], [Handley03]), but this is conceptually based on matching the behaviour of a particular transport protocol. This comes down to maximising a default utility objective, which may not be the one best suited to the application with semi-elastic demand for bandwidth. Such applications would benefit from a greater degree of policy freedom (and hence delegation of control).

The framework above is loose about exactly what information is captured by *p*. In reality, even when considering only the forwarding plane, a network integrates many different node types, from wireless relays to large scale routers and even optical wavelength switches, with many different underlying 'physical' resources – that is, elements which require investment and maintenance, and hence whose load it is an economic requirement to manage. Any attempt to develop a taxonomy for such resources would be instantly obsoleted by technological advances, and cannot be the basis for a future Internet architecture. Instead, we have carried out an initial analysis of the types of resources that currently exist, and how they are depleted by forwarding plane activity. In accordance with the Design Principle of Section 2.1, we only attempt to signal metrics that correspond to forwarding plane concepts, namely packet transmission. At this level, it appears that all resource usage can be reduced to at most three dimensions:

1. Depletion caused by packet rate (for example, forwarding engine lookup, medium access control overhead, header RAM storage).
2. Depletion caused by bit rate (spectrum, line capacity, battery drain).
3. Depletion caused by traffic 'burstiness' (packet buffering).

The distinction between the first and second is analysed in more detail in [Briscoe08]; in the current Internet it appears that the second is far more significant (i.e. more likely to be congested in practice), but in principle the two measures are both required. Whether burstiness (or indeed even higher order traffic statistics) needs to be measured is still an open question in our work. In the following, we therefore assume that all resource loading can be reduced to a single (multi-part) metric *p*; the way in which *p* is calculated for each link in the network depends on the equipment type and is a local matter for the equipment and network designer (in accordance with the overall framework above). However, exactly how many parts are really needed in practice remains a research topic, especially given the cost of signalling large amounts of resource information. This situation is summarised in Figure 7 below. We also leave open the precision with which *p* has to be signalled. In the current Internet, essentially only a single bit is available for ECN and so an accurate value of *p* has to be integrated at the receiver over many packets. We research the benefits and implementation possibilities of techniques such as [Andrew06] and [Pan07] using multibit congestion control fields, which impose lesser requirements on path consistency and decrease the time needed to respond to changes.

Finally, we emphasise that the above problem decomposition and its supporting signalling mechanism work initially in a network based on mutual trust, where nodes are always truthful about the load they experience, packet sinks report the information reliably, and sources adapt their sending rates accordingly. How to guarantee these properties, whether by enforcement or incentivisation is clearly non-trivial; this fundamental problem is explored in Section 3.3.4.

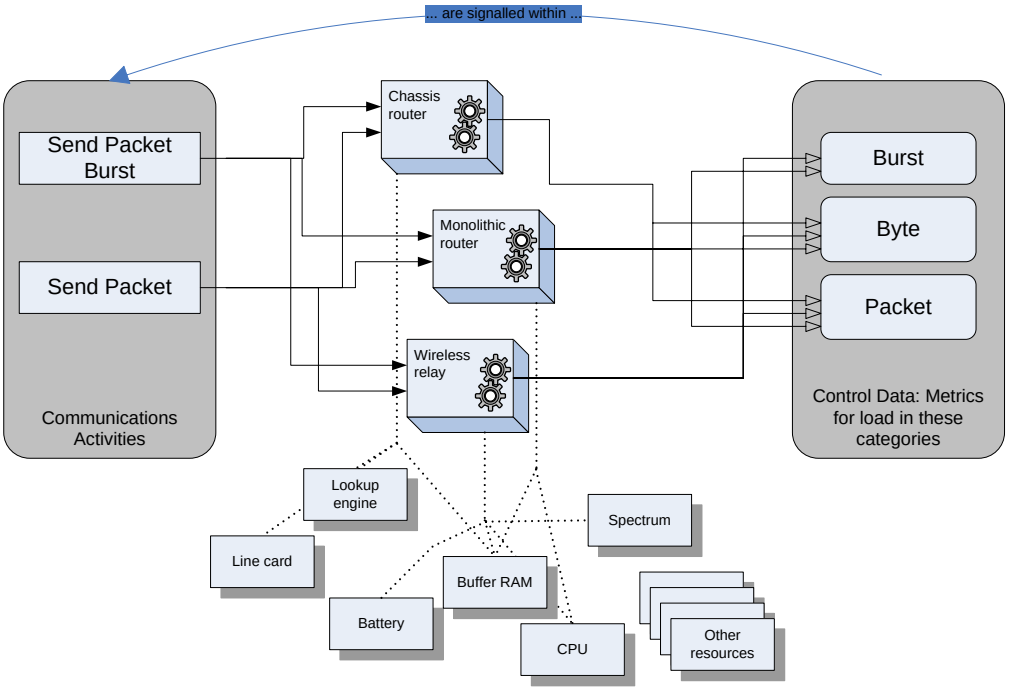

Figure 7: Model of Transmission Resources

3.3.2 End-to-end Multipath

The presentation so far has left open the question of how the transport services invoke the underlying packet delivery functionality. The simplest case, which matches the transport protocols defined in the Internet today, is that each end-to-end connection corresponds to a single instance of a resource control loop in the forwarding plane; and consequently, the packet flow exercises a single path at any given time. However, this property is not intrinsic to the architecture.

Using multiple paths to support a single end-to-end connection is a concrete example of the resource pooling principle, as discussed in Section 2, and general motivations are discussed there. Resource pooling for transmission paths provides a number of obvious benefits in terms of improved resilience and capacity (both for individual flows, and in terms of the overall admissible set of flows as a whole) which are common to any resource pooling mechanism. An additional benefit, that arises when resource pooling and adaptive congestion control are appropriately combined, is that resource network providers and consumers can automatically achieve load sensitive traffic engineering, including for inbound traffic, without active management of the reachability plane. Further discussion of the motivations and analysis of resource pooling in the Internet are given for example in [Handley08]; in this Section, we explain how this functionality is accommodated in the Trilogy architecture.

End-to-end multipath has implications for all three architectural building blocks, although it does not significantly increase the binding between them:

- In the reachability plane, it must be possible to utilise different paths between the same pair of endpoints. Several techniques are possible: the simplest is where a destination (and possibly also the source) has multiple locators. Each locator pair corresponds potentially to a different path, in at least some part of the network. Even if multiple locators are not available, we assume that varying the path selectors (see Section 3.1.1) will exercise multiple paths, provided the underlying routing system can maintain multiple routes for a single destination.
- In the forwarding plane, the queue management and packet marking within the network are unchanged (i.e. we do not change the property that packets are self-describing). The operation

of path estimation and feedback from the receiver are also unchanged; note that the receiver treats each path as being independent. The sender is now responsible for outgoing path selection as well as packet scheduling in time, according to locally defined policies as before.

- Responsibility for end-to-end properties remains part of the transport service. That now has to deal with the performance impact of packets arriving over multiple paths (e.g. lack of ordering preservation).

Note that the multipath capability is invoked implicitly by the source formatting packets (e.g. by address selection or path selector setting) so that they can be diversely routed by the network. The receiver must also understand how packets correlate with paths (in order to carry out path estimation and feedback). This requires some formal definition of exactly what identifiers can influence route selection, in other words, a standardisation of the service provided by the reachability plane.

Single path resource pooling

Multi-path resource pooling

Figure 8: Resource Pooling over Single or Multiple Bit Pipes

There is no requirement on the network to provide disjoint paths (either end to end or over particular segments), and no way for end systems to explicitly request such routing: we wish to keep end systems and network operations decoupled. In other words, a source cannot request that packets follow different paths; it can only offer that option to the network, and request that particular subsets of packets should be consistently routed. There are several open questions about the level of path diversity in the network: [Mitzenmacher01] suggests that it is only necessary to offer a small subset of the possibilities, whereas [Godfrey08] indicates that choosing the wrong subset can lead to very poor results. Our future work includes developing techniques to expose multiple paths in routing protocols such as BGP, and measuring the usable diversity that results for typical Internet provider topologies.

For the resource control algorithms, we leave open that the control loops for a single end-to-end connection may be coupled. Informally, if a source is multipath capable, it would be attractive for its traffic to be moved more aggressively off a congested path onto an alternative route, and this is the effect of algorithms such as [Kelly05] and [Key07]. This means that the forwarding plane needs to understand which traffic streams are related in this way, which must be indicated from the transport services. End systems are therefore likely to couple the transport and forwarding plane implementations for this and other reasons (e.g. intelligent handling of re-ordering); however, we can keep the identifier spaces of each layer conceptually separated.

3.3.3 Alternative Reachability Architectures

The current Internet architecture confuses the concepts of locators and identities with IP addresses effectively acting as both. The latter is really about identification of the communicating parties

(entities) whereas the former is about selection of the communications path. This confusion of concepts has led to a shortage of global IP addresses, as well as knock-on consequences for capabilities like traffic engineering, multihoming and provider independence [Meyer07].

In turn much, research has been done looking at alternative routing architectures which seek to separate the two concepts, for example LISP, Six/One and Shim6 ([Farinacci08], [Vogt07], [Nordmark08]). However, people use the terms locator and identifier to mean a number of similar but not identical things [Halpern08] – for example, whether an address names an IP interface or effectively also names the provider via which to reach the interface. Another distinction, analysed in [Zhang08], is whether the impact of the locator-ID split is to separate out edge prefixes from the transit core (e.g. LISP, Six/One) (which requires a mapping system) or to push multiple provider aggregated addresses into the hosts of multihomed sites (e.g. Shim6 and the approach of Section 3.3.2). Except for the approach of Section 3.3.2, all the other current proposals are based on the current Internet architecture and don't map cleanly into the Trilogy architecture.

At the most essential level we follow the approach of [Saltzer93]: applications need globally recognisable names (identifiers), but they do not need to know how the lower layer finds the entity represented by that name. In terms of the Trilogy architecture, this means that a name is passed down by the transport services (Section 3.1.4) and a lookup, managed by the reachability plane (Section 3.1.1), maps it to an address. The address is internal to the reachability plane and is used to build routing systems.

The nice thing about this model is that it allows a clear separation of functions between identifying applications on a given end-host and locating that end-host on an appropriate network (assuming that in future many hosts will be multi-homed). This type of locator-ID split helps maintain the separation of functions (see Section 3.3.5). For instance it supports the 'fuzzy ends' Design Principle: if a function hasn't been delegated by the end point to the network, then the end point could obfuscate the original identifier so making it hard for the network to interpose itself in the communications.

Beyond this, an open issue is how to handle recursive networks (networks over networks over networks…). [Day07] suggests that rather than the current layered model (application, transport, IP, networking technology) instead layers are defined by their administrative scope (e.g. VPN, peer to peer network, LAN, AS). We are exploring how to apply this scoping idea to Trilogy's architecture of reachability plane, forwarding plane and transport services.

3.3.4 Accountability

When several users share a common resource, such as the Internet, they are accountable to each other every time they consume part of it, because they may block competing users who would value the privilege of using that same part of the resource. The congestion a packet experiences on its end-to-end path has been shown [Kelly98] to quantify the extent to which it is accountable for the degradation of the network service of every other packet – and by extension to the service degradation of all the traffic that is using the network at the same time.

Note that the following discussion focuses on accountability for real-time usage of the network's shared resources, not accountability for the cost of deploying the resource in the first place (initial connection charges). These two may be bundled together in places, but they don't have to be. In our future work, we will also consider accountability for other aspects of network operations, such as impact on global routing (BGP churn); however, the real-time resource accountability case is the one best understood at present.

If each individual packet is held accountable for its usage of the shared resource, the important question is to figure out where accountability enforcement is best done. The point of that function is to arbitrate the overall allocation of network resources, to ensure that all stakeholders get a share of the common resource that is sensible with respect to the network service they have paid for (given the

dynamic usage of the same network resources by all other stakeholders). This suggests that accountability would be best policed at the level of connectivity contracts.

In the case of multi-homed customers, this means that there is a need to police each connectivity contract independently. For instance, Figure 9 depicts a scenario where customer C who hosts users u_1, u_2 and u_3 is multi-homed to access providers AP_1 ad AP_2. In that case, customer C is accountable to AP_1 for flows f_1 and f_2, while it is accountable to AP_3 for flows f_3 and f_4.

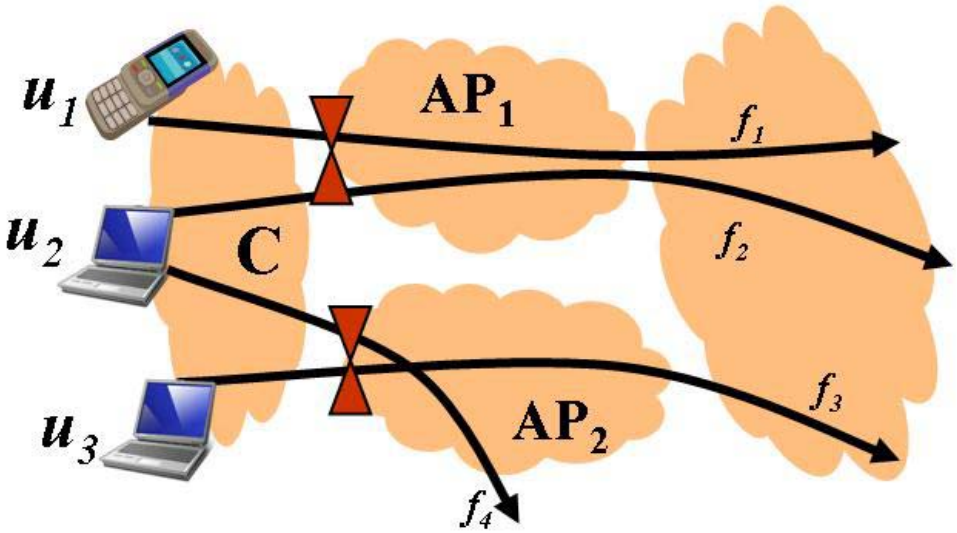

Figure 9: Accountability of multi-homed customers monitored at each access connection

Further research is needed to identify how practical implementations of accountability enforcement can be realised without compromising the separation of naming and resource control. In particular, we need to analyse how the accountability enforcement system discriminates between different stakeholders. Initial analysis suggests that connectivity contracts neither can nor need to be associated with the identifier spaces already identified as part of the baseline architecture, with the exception of service class identifiers. For instance, a corporate network would be accountable for all the traffic its end hosts send into the public network.

Accountability for using shared resources remains a complementary function to end-to-end per-flow rate control: a stakeholder held accountable for its usage of the network still needs to decide how to control the rate of the traffic generated by the application(s) of the all user(s) it represents, but that is a matter of internal policy for each stakeholder.

3.3.5 Functional Distributions

The host/router model described in Section 3.2 represents one possible packaging of functionality into nodes and networks, but it is not by any means mandated by the architecture. For example, we make no assumptions about how end-system and packet delivery functions are organised into economic entities: organisations traditionally classified as network operators can be expected to deploy applications as part of their overall service offering, and users will also operate their own networks (for example, in corporate or home network environments). Even in networks which purely offer a packet delivery service externally, nodes will include endpoint functions for reasons such as network management.

Distribution of application layer functions over the infrastructure can be seen as one (weak) application of the 'fuzzy end-to-end' principle (Section 2.3), allowing network service providers to take some control from the end system without violating the basic architectural principles. Note that, at this stage, this is limited to explicitly participating applications and the transport services that support them – in other words, interception of arbitrary application layer traffic is not allowed. (Indeed, given that application traffic type is not visible to the transport or packet delivery service, application specific interception should in theory be impossible. Instead, application designers must specifically design this distribution into their system architectures, for example by supporting proxy modes of operation, and selection of 'in-network' application endpoints must be done via network management or modification of whatever locator lookup mechanism is used to find application peers.)

In the Internet architecture, the IP layer itself is not divided. Our decomposition of the packet delivery service has planes for reachability and forwarding, and these can in principle be implemented separately; the identifier spaces relevant to each are distinguished partly for this reason. Some nodes perform outgoing link selection in regimes where there is no need for active queue management at the per-packet level (for example, in future optical networks). The converse case is nodes which might have trivial routing behaviour, or have the physical routing determined by link layer technologies, but still carry out traffic management in the forwarding plane[12]. Our architecture allows the direct use of the forwarding plane component at such locations, as an alternative to requiring explicit interworking between network and link-level load control mechanisms. This splitting of the packet delivery service can in fact be seen as a very simple example of a role-based protocol architecture [Braden03], where the roles (i.e. planes) can be deployed independently or in combination.

3.4 Potential Extensions

Needless to say, there are many proposals about future network architectures which do not fit directly into the minimal architecture presented above. These fall mainly into two groups:

- Modified identifier/locator schemes (as discussed in Section 3.3.3 above);
- Packet filtering or pre-authorisation schemes, mainly targeted at the problem of distributed denial of service attacks ([Anderson03], [Handley04], [Ioannidis02] [Yaar03], [Yaar04], [Yang05]).

These are not just theoretical possibilities: corresponding functions exist in current networks, for example for NAT traversal in the first category, or firewall traversal in the second. The range of techniques for each is large and growing, ranging from dynamic discovery and reverse-engineering of middlebox properties (such as STUN [Rosenberg03]), through localised network management systems ([UPnP06]), all the way to fully fledged control plane protocols ([Stiemerling08]).

Some of the requirements met by these schemes relate to concepts already in the architecture. For example, we have been careful to distinguish the identifier spaces used for naming transport and application endpoints from the space used for locators, and we have also identified the presence of policy controls in the reachability plane. In so far as identifier/locator management and (D)DoS protection are concerned, where these can be performed entirely at the end system by appropriate implementation of the transport and packet delivery services there, the architecture is already sufficient.

The difficulties arise where implementation at the end system is not possible or not optimal. There may be many reasons for this:

- The need to support existing (legacy) hosts.

[12] It is notable that in several residential broadband technologies, the natural location for congestion management (the head-end modem in cable systems, or the DSLAM in ADSL) is not itself an IP router; this limitation has apparently significant consequences for P2P traffic management.

- Scalability: for example, to carry out locator/identifier management for whole groups of hosts, shielding the rest of the Internet from per-host operations.
- The need to impose a policy (e.g. on security or network access) at a higher level than the host level, for example in enterprise networks. (This can be seen as a special case of scalability: carrying out management and audit for individual hosts may be impractical or impossible for organisational reasons.)
- The fact that the function involves not only the endpoints but also the network itself. For example, mobile hosts may wish to filter traffic *before* the last (wireless) hop, to save transmission costs (battery drain and service charges). In the case of DDoS attacks, the main victim of the attack may be a congested link deep within the network, which end systems cannot protect. Ideally DoS traffic should be filtered as near to its source as possible, rather than at the destination, and yet only the destination is fully aware that the traffic is malicious.
- Limited knowledge: the network may know about things like topology or routing policy which the end point cannot know about, e.g. due to BGP's one AS-hop horizon.

The critical feature of all the proposed mechanisms to date is that they require some additional state in the network, where that state depends in some way on the current state of communications between two or more nodes. The nature of the state varies – it might be some kind of identifier mapping, or opening or closing of a firewall pinhole, for example. Regardless of the details, the existence of the state in the network violates the principle that underlies the forwarding plane that packets are self describing. State can obviously be installed in the reachability plane; however, the scalability constraints of the reachability plane make the concept of host-related state (especially state dependent on communication status) intrinsically hard to accommodate.

A further issue arises because of the nature of the functionality involved. An underlying principle was that state should be soft and hinty: loss of state should at most degrade the network performance, not break it entirely. However, loss of identifier mappings would typically break communications; loss of security information would usually be unacceptable unless the network was configured in a default-deny fashion, but that then requires explicit state setup to achieve any communication at all.

The project does not currently have a uniform architectural approach to incorporating such functionality in to the architecture in a scalable and modular fashion. However, we have a number of techniques under consideration. One of these is to extend the accountability mechanisms that support the forwarding plane (see 3.3.4 and [Briscoe05]) with capability-style functionality ([Anderson03]), to police DDoS attacks in the same way as congestion. Another is to look at more radical distribution of what have traditionally been considered endpoint functions (destination locator lookup, transport layer aspects) over multiple nodes. An example is to delegate the responsibility for participating in the global reachability plane to a remote device, so nodes within a network can use locally defined reachability mechanisms. Both of these techniques can be seen as applications of the fuzzy ends principle (Section 2.3), and our future work will attempt to extend this paradigm to other areas, possibly feeding back into a refinement of the baseline architectural components themselves.

3.5 An Integrated Control Architecture?

One of the goals of the Trilogy work is to develop a high-level framework for reachability and resource control, including their interactions. This should not be confused with attempts to simply integrate the reachablity and resource control functions of the Internet. In this Section, we revisit this question and evaluate to what extent our framework covers these interactions.

We have not attempted to develop a tightly integrated architecture, where a single protocol manages both aspects of network operation. Such approaches were developed in the ATM world (for example PNNI [ATM-PNNI96]), but they conflict with our deeper goals for modularity and simplicity. Superficially, the baseline described in Sections 3.1 to 3.3 looks even less integrated than the current Internet, because of the separation of the reachability and forwarding planes.

However, integration is not an end in itself – the purpose is to enable networks to develop and evolve in each area, without being hampered by unexpected feature interactions. We have tried to be explicit about exactly how and where reachability and resource control interact, in terms of use of common identifier spaces and assumptions about path consistency. This basic decomposition allows us to include new integrated network functions as a set of cooperating refinements of each building block. Our first example is end-to-end multipath, which has implications for all of the reachability, forwarding and transport components, but very limited changes to their interactions. We hope that the same basic approach will serve for other functions which naturally straddle the reachability and resource control boundary, such as DDoS protection and accountability enforcement, and this will be one of the main criteria in evaluating the architecture extensions of Section 3.4.

4 Economic and Social Interactions

A key goal of the Trilogy project is to define an Internet control architecture that is adaptable to the socio-economic context at which it is targeted. Many aspects are important, but in this section we pay particular attention to the socio-economic 'tussles' that take place between the various stakeholders in today's commercial Internet. The idea of 'tussle' between Internet stakeholders is a concept outlined in [Clark05]. The underlying premise is that the Internet, originally designed by, and for, a small not-for-profit academic user base has grown to become an all-pervasive commercial entity with many competing socio-economic interests – the 'tussle' – that requires the adoption of new or modified goals and principles to be applied to the process of Internet architecture design. Furthermore, the control architecture should be made as future-proof as possible so that other as yet undefined tussles can be played out between as yet undefined stakeholders.

The purpose of this Section is to show how the Trilogy Design Principles introduced in Section 2 take account of this wider socio-economic context. This has been done by adopting an analysis framework to trace the linkages between the Internet's socio-economic goals and the corresponding technical design goals and architecture Design Principles. Our approach has been to take the technical design goals and architecture Design Principles of the original Internet architecture, and reconsider them in the context of the socio-economic goals and 'tussles' of today's commercial Internet.

4.1 The Original Technical Design Goals and Principles

The original Technical Design Goals of the Internet as stated in [Clark88] are as follows:-

1. **Interconnection of Existing Networks:** This design goal was about interconnecting isolated networks of the time, rather than imposing a new unified global network that could become obsolete later.
2. **Survivability:** This design goal was about ensuring that Internet communication services could continue even when some components – networks, gateways – fail. In particular at the time of the original design this was interpreted to mean that after a temporary failure the entities communicating should be able to continue without having to re-establish or reset the high level state of their conversation.
3. **Support for multiple Communication Service Types:** This design goal stated that the Internet should support, at the transport layer, a variety of applications that could be distinguished by differing requirements for such things as bandwidth, latency and reliability.
4. **Support for a variety of Physical Networks:** This design goal was that the Internet architecture should operate over a wide variety of physical networks by making a minimum set of assumptions about the function provided by the network layer.
5. **Distributed Management:** This design goal identified that Internet resources should be able to be operated and managed by distributed stakeholders.
6. **Cost Effectiveness:** This design goal stated the desire for efficient use of network resources.
7. **Simple Host Attachment:** This design goal stated the desire for keeping the complexity of host protocol stacks low, so that deployment was not hindered.
8. **Resource Accountability:** This design goal stated the desire for understanding and monitoring the usage of network resources.

This is an ordered list and, as stated in [Clark88], this was crucial since "an entirely different network architecture would result if the order were changed" [Clark88]. It is also clear from [Clark88] that the original Internet Architecture primarily took account of Goals 1 – 5 only. Indeed, some of the architectural design decisions related to these five goals led to compromises on some of the lower priority goals (numbers 6 – 8).

These design goals led to the basic Internet architecture, i.e. a connectionless packet switched communications network consisting of endpoints (hosts) connected by intermediate packet switching nodes (gateways), and four supporting architecture Design Principles, as follows:

- Self-Describing Datagram, the principle that control information should be carried within each datagram including any address / identifier information necessary to route datagrams between endpoints;
- Fate Sharing, the principle that state information should be stored within the entity where it is used, e.g. per-flow state should be kept at endpoints, with only state required for packet processing (the operation of the network) stored in the network;[13]
- Layering, the principle that a hierarchical protocol structure with a clear identification and separation of the services provided by each layer to the layer above. In particular, the network layer is the unifying layer, supporting the heterogeneity of the higher and lower layers, i.e. 'IP over everything and everything over IP';
- End-to-End Argument, the principle that "certain required end-to-end functions should only be performed by the endpoints (although supporting low level mechanisms are justified if they provide performance enhancements)" [Saltzer84][14].

Arguably, a further architecture Design Principle can be identified as follows;

- Distributed Control, the principle that autonomous entities within the Internet should be responsible for management of their local policy and data[15].

The analysis framework used to identify this set of original Technical Design Principles is summarised in Table 2 in Annex A. The second row in the table consists of the eight technical design goals identified in [Clark88], in order from left to right. The first row contains some higher level 'stakeholder' goals, which we believe to have played a large part in determining these original design goals. The third row captures the fundamental design decisions that drove the architecture in a particular direction, the fourth row highlights some of the architectural implications of those decisions and the fifth row identifies the architecture Design Principles.

The table is best read column by column, from left to right, since some of the lower priority design goals were influenced by decisions and principles adopted in satisfying those with higher priority.

4.2 A Set of Architectural Principles for Today's Internet

Designing an optimised architecture for today's Internet would undoubtedly have led to different design decisions being made with, we believe, a greater emphasis placed on the achievement of design goals 5 – 8 since these encompass most aspects of today's socio-economic tussles. In fact, one might say that one of our tasks within the Trilogy project is to bring these aspects into the mainstream of Internet architecture design in order to incorporate 'design for tussle'.

[13] This avoids the need to provide mechanisms to discover and reinitiate remote state information in the event of a failure.

[14] This principle only received its definitive formulation in [Saltzer84] but was clearly an original technical Design Principle, a logical extension outcome of the fate sharing and layering principles.

[15] Examples are the Domain Name System (DNS) which allows domain owners to manage all entities within that hierarchy; and inter-domain routing, where each Autonomous System (AS) is responsible for internal management of its own resources, with exchange of information in a common format (which leads to the two-tiered routing system of intra- and inter-domain routing).

So, rather than starting with an ordered set of eight technical design goals, we begin instead with the following (non-ordered) set of eight socio-economic goals.

- **Global Connectivity:** This socio-economic goal is about providing the ability for end users to communicate with one another via the Internet, for both business and social purposes.
- **Business Continuity (and Enhanced User Experience):** This goal reflects the increased importance of the Internet as a business (and social) tool, with high demands on resilience and resource availability.
- **Application Innovation:** This goal is about encouraging innovation in the development of applications that will operate over the defined network architecture.
- **Network Heterogeneity:** Here, the goal is to encourage flexibility and ubiquity of Internet access – independent of the underlying network access technology.
- **Business Autonomy:** The goal here is to ensure that Internet businesses exist in a competitive market where they can enjoy sufficient autonomy and incentives to invest in network infrastructure, application innovation and service differentiation.
- **Utility Maximisation:** This goal aims to ensure the efficient use of scarce networking resources in order to maximise the utility of end users.
- **Accessibility:** This goal is about facilitating simple access to the Internet and its services, from both a technical and market perspective.
- **Accountability:** This goal requires that an accurate and traceable record is available regarding the supply and usage of Internet resources.

Our analysis for the identification of a new set of architecture Design Principles based on these socio-economic goals is captured within Table 3 in Annex A. The general format of the table is the same as Table A but takes as its starting point this set stakeholder goals that reflects the socio-economic needs and expectations of *today's* Internet stakeholders. Rows 2 and 3 map the following tussles, identified during the development of [Trilogy08], to these socio-economic goals.

- The tussle between end users (including content/service providers) and ISPs over the price and terms of **Network Access**
- The tussle between end users and content/service providers over the price and terms of **Content/Service Access**
- The tussle between end users (and also involving regulators) over **End-to-end Connectivity** between hosts
- The tussle between ISPs over the control of **Routing** within the network
- The tussle between end users (typically policed by ISPs) over the allocation of **Network Capacity**

It should be noted that some of these socio-economic tussles are best resolved outside of the Internet architecture, in which case it is the role of architects like ourselves to avoid unduly biasing the architectural solution towards one stakeholder group over another. A particular example of this is the Accessibility goal, relating to both Network and Content/Service Provider Access, where we believe the tussles are best resolved through the creation and maintenance of a competitive market.

Working down the columns of Table 3, the transition is shown from these design goals to the architectural implications, and the final row shows how these goals map to architectural Design Principles. Some remain unchanged from the original principles as shown in Table 2, some are a modification, and some are entirely new. This is not surprising given the nature of today's Internet in comparison with that for which the original Internet was designed. Rather than being a separate set,

however, these original Design Principles can be shown to be subsumed into our new principles. The strongest original principles, as identified earlier, were:

- The self-describing datagram
- Fate sharing
- Layering
- End-to-end
- Distributed control

The self-describing datagram principle remains a fundamental component of Internet design, and the Information Exposure principle can be seen as an extension of the original principle by suggesting that the action of consuming resources should be integrated with a metric indicating the resource usage. 'Fuzzy end-to-end' can be seen as a natural evolution of the original end-to-end principle, whilst Fate sharing and Layering can also be subsumed into this principle. The Distributed Control principle can be restated as the 'separation of policy and mechanism' principle, with elements of the delegation of control picked up in 'fuzzy end-to-end', leaving the four principles described in more detail in Section 2:

- *Information exposure principle,* suggesting that sufficient information about resource usage should be exposed to support an effective and efficient allocation of that resource
- *Separation of policy and mechanism principle*
- *Fuzzy end-to-end principle,* suggesting that the endpoints explicitly delegate some functions to the network
- *Resource pooling,* suggesting that resources in the network should be able to be pooled in order to improve effectiveness and efficiency of the network

4.3 Supporting Design for Tussle

A key question is whether our proposed architectural Design Principles are able to accommodate the tussles identified above. It is our view that they can.

In fact, as already stated, it is our belief that the Network Access and Content/Service Access tussles are best resolved by market competition between suppliers (ISPs and content/service providers), with end users having a fair and open ability to make informed choices. This means therefore that there are no architecture Design Principles associated with these particular tussles, other than the need to avoid undue bias in the architecture towards any one stakeholder group. In the Network Access tussle, it is the role of regulators to ensure that there is competition between ISPs, allowing end users to switch between them with minimum cost. To support this, the architecture must be neutral and effectively modularised by, for example, decoupling identity from location to reduce provider lock-in through provider-assigned addresses (which would tend to impose a high switching cost on end users). Within this (regulated) competitive market for network access, it should also be possible for content/service providers to compete fairly and openly for end user business.

The tussle over end-to-end connectivity is broadly about end users desiring the ability to control who they connect to, and who can connect to them. This can both take the form of restricting access (e.g. firewalls) as well as enhancing access (e.g. making use of multiple paths). As such, this relates to at least three of the newly defined Design Principles: **'resource pooling'**, for connection enhancement; '**separation of policy and mechanism'**, so that various stakeholders can express their preferences independently; and '**fuzzy end-to-end'**, allowing stakeholders to delegate their control over end-to-end functionality if desired.

The tussle over routing has both technical and socio-economic aspects – not only is it about providing global connectivity, but the business arrangements that support this have many nuances. For example, autonomous systems may wish to limit the peering provided to neighbours, configure specific backup links, or avoid some traffic altogether. The **'separation of policy and mechanism'** principle here enables this business autonomy to be applied without affecting the mechanism by which routing is provided.

The tussle over network capacity is over a clearly defined scarce resource, and as such some mechanism is required to allocate it efficiently between end users. In our view this should be achieved via a market mechanism that internalises the externalities, although whether this sits within the architecture or outside of it is an open issue. This tussle is supported by the 'i**nformation exposure'** principle, which, due to the many implementation options, is expressed in terms of exposing the necessary information regarding the costs of resource usage in order to support a market-based mechanism.

The network capacity tussle is notable for some significant dependencies between stakeholders which make it more difficult to isolate the tussle. Firstly, network capacity is provided by ISPs but consumed by end users and content/service providers. Secondly, in general, an individual ISP does not have full control over the end to end path and, therefore, provision of network capacity to an end user or content/service provider. Instead it is dependent upon the entire chain of ISPs between the end user and the content/service provider. The key implication of these dependencies is that no one stakeholder will have complete control over resources on a path, and thus will not be able to have unilateral influence over how packet forwarding will behave. This provides a challenge to the architecture designer, in order to support the numerous tussles that will exist between stakeholders both along, and outside of, this path.

Finally in this section, how do the initial set of 'principles' that emerged from the 'Design for Tussle' Case Studies [Trilogy08] relate to this discussion? This is best done by mapping the principles from that work to their three core groups:

- Isolating tussle (through modularisation and defined boundaries)
- Managing control (allowing control to be flexibly allocated and distributed)
- Economic mechanisms (provide support for market mechanisms, with alignment of costs and benefits in time and space)

A more accurate term for these 'principles' would probably be intermediate 'goals', as they provided a stepping-stone towards concrete Design Principles. Indeed, as can be seen from the preceding discussion, all of the aspects related to each of the three core groups are clearly present in our proposed Design Principles.

For example, the isolation of tussle is an important factor within the principle of separation of policy and mechanism, i.e. allowing stakeholders to apply their preferences internally, while playing out tussles by interacting over a common mechanism. The management of control is integral to 'separation of policy from mechanism' (by ensuring that stakeholders can manage their own policy separate from external interactions), and to 'fuzzy end-to-end' (by allowing control to be distributed and delegated as required). Economic mechanisms can be applied across all principles; however the most relevant is the information exposure principle: the exposure of information regarding the usage of resources is a prerequisite for the application of economic mechanisms in order to fairly allocate this resource.

5 Technical Analysis

This Section applies the Design Principles defined in Section 2 to resource control and locator/identifier separation.

5.1 Resource Control

Currently most resource control in the Internet is done using variants of the TCP protocol (though other protocols such as TFRC and RTP/RTCP also exist. TCP is essentially a window-based loss-driven rate control protocol: it allows a certain number of unacknowledged bytes to be in flight at any one time; if there is a loss-event this number is drastically reduced while it slowly increases if there is no loss. We also consider alternative proposals such as:

- eXplicit Congestion control Protocol (XCP [Katabi02]), where routers provide explicit feedback to hosts about how they should change their sending rate to achieve high utilisation and fairness between flows;
- Rate Control Protocol (RCP [Dukkipati05]), where an explicitly signal from the routers lets each source know what its maximum fair throughput is;
- TCP Multipath, a multipath version of TCP where the TCP congestion control algorithm is applied to the aggregate throughput of each flow. The source has visibility of several parallel paths to the destination and may choose to split the traffic load it generates across several of these paths.

5.1.1 Information exposure principle

End-to-end rate control mechanisms use different information signals which are implicitly or explicitly integrated in the communication. In the early Internet, routers only implemented Drop Tail queues: excess packets are dropped when the buffer is full. This provided an implicit congestion signal back to at least some of the sources sending traffic across congested bottlenecks which allowed responsive applications to back off. However, the signal only occurs once the resource is already overloaded and isn't suitable for preventive rate adaptation to avoid complete saturation of the bottlenecks. This mechanism also suffers from binary granularity, i.e. a packet is either lost or it is not. Additionally, packet loss is an overloaded signal, i.e. corruption packet loss may be mistaken for congestion.

Active Queue Management (AQM) using algorithms such as Random Early Detection (RED) was introduced in the early to mid 1990s to alleviate that problem and to prevent the phenomenon of global synchronisation where large numbers of sources simultaneously backed off in the presence of congestion. With AQM, packets can be probabilistically dropped before the buffer is completely full, which allows sources to react to incipient congestion, before the congested resource is completely overloaded.

As the resource is now able to report imminent congestion before it actually happens there is no real need to drop packets. ECN marking was introduced to replace the drop by the equivalent of an in-band flag, which can be detected more quickly than a lost packet (a TCP source requires the acknowledgment of several subsequent packets before it detects a loss and acts on it). Several proposals have been made to increase the granularity of the coding of the congestion level [Andrew06, Pan07] which would improve the accuracy and timeliness of the congestion signal.

For a large proportion of the traffic, rate adaptation is deferred to TCP, whose rate control operates on explicit and implicit congestion signals. In most adaptive applications, the signal driving the rate control is the congestion level (whether implicit or explicit). Other proposals suggest signalling metrics about the rate of the flow: for instance routers may signal the fair rate the flow should use (RCP) or the rate adjustment required to reach it (XCP).

Although all these control frameworks include a control information loop in-band with the date, congestion signalling captures better the externality caused by each packet on other packets (thus by

each flow on other flows, by each user on other users, and by each network on other networks). Another advantage of this approach is that it doesn't require bottleneck links to make any decisions on fairness or rate control.

The congestion signal can be used to implement a congestion priced network service, which is the most efficient way to ensure all stakeholders optimise their objective [Kelly98]. It also allows the accountability of any stakeholder to be defined with respect to the externality the traffic they forward has on other stakeholders.

Finally, it could be important to discriminate between network layer congestion (e.g. router congestion due to excess demand) and lower layers congestion (e.g. packet corruption due to interference) if the two problems are better addressed by different parties of the communications system.

5.1.2 Separation of policy and mechanism principle

The goal of modularity is to separate local policy choices (i.e. those that are implementation-specific) and the external common protocol (i.e. that which is defined by standards). In terms of 'classic' TCP, resource control is operated effectively at the end-systems based on the congestion signal it receives from the network. As the TCP protocol embeds in its design the specific rate control mechanism that needs to be used, it presents very limited modularity to the end user.

XCP and RCP also embed in their design an inflexible assumption about how the fairness policy will be imposed, as a congested bottleneck will aim to assign an equal share of its bandwidth to every flow, which then sets the rate sources are able to use.

To get a more modular resource control framework, we need to consider a network where routers signal congestion based only on the state of the bandwidth resource they control; sources are not limited as to which rate control algorithm they should use by a decision that has been taken for them at bottlenecks elsewhere in the network. In that context, if users are kept accountable for the amount of congestion they create in the network they are able to choose how to adjust the rate of all the traffic they generate such that the congestion created by the aggregate stays within their means (a congestion cap, or a budget cap if direct congestion pricing is used).

Examples of how different policies can be deployed for different types of users based on the same congestion information can be found in [Gibbens99].

5.1.3 Fuzzy end-to-end principle

Applying a congestion cap to a customer leaves them more freedom than insisting that all flows to TCP-friendly, since users are free to decide how to allocate their congestion cap between different applications. This means the control decision shifts entirely to the end-user.

In some cases the end-user may not be able to handle the resource control, e.g. due to lack of knowledge, lack of appropriate APIs, lack of accountability etc. Hence, the user may choose to delegate some of the resource control functionality to the network. For example, within a corporate network, the administrators may take some resource control decisions on behalf of the employees/hosts. If the corporate customer is submitted to an aggregate congestion cap, the administrator may decide how the cap may be split between different users. The control decision would therefore be shared between the local network administrator and the end-users.

5.1.4 Resource pooling principle

The current Internet architecture has completely decoupled completely reachability and resource control. Routing decisions are made by network operators based on local policy and on the outcome of routing protocols. Subsequently, end-systems send traffic to destinations based on a route that has been predefined. Resource control happens at the end systems for each flow over each route.

Resource pooling is intended to maximise the usage of a network by pooling all its resources into a single virtual resource. The concept of resource pooling is to allow network load to be relocated or spread across network resources. Today, suboptimal resource pooling is sometimes provided by higher-layer applications in bespoke circumstances, such as content delivery networks (CDNs), and peer-to-peer applications. CDNs operate a variety of mechanisms by which load can be shared between widely distributed servers. The choice of server is dynamic and transparent to the end user, however. In peer-to-peer file-sharing, the content is available from several distinct sources, and destination clients therefore have a choice of paths across which they can retrieve the data. This spreads the load on the network and in effect pools the resources (network paths and actual file data)

The proposed TCP multipath variant presents the arrangement the other way round: one source gets visibility of several paths to reach a destination and is able to balance its load between these paths in response to the path conditions.

Currently TCP doesn't benefit at all from resource pooling in terms of multiple paths. It is given a path by the routing system and operates resource control on top of that path. It could get more exposure to resource pooling if several path were presented at the start of a connection. This would be a degenerate version of TCP multipath where there is ultimately one path used, which would be chosen by the end-system. However there is no current mechanism by which the end-node is able to tell the current path conditions for those paths and thus any choice would be relatively arbitrary. In contrast, classic TCP leaves the choice of the route to the network operators.

5.1.5 Conclusions of the analysis

The Trilogy Design Principles are very pertinent to the issue of network resource allocation in the Internet. Most significantly, they help assess how design choices affect the balance of control between the stakeholders. They also help detect design choices that strongly embed policy choices as integral parts of the mechanism, which would be detrimental to the long-term evolvability of the architecture.

For instance, proposals such as XCP and RCP embed inadequate information exposure that limit the range of fairness policies that can be used, and prevent local policy variations. The resource pooling principle is particularly key to widening the degrees of freedom users could have with respect to making control decision about the network service provided to their traffic.

Further research is needed within the Trilogy project and the broader research community:

- to better specify what form the information exposure mechanism should take (how many metrics need monitoring? where are the information signals needed? …)
- to define the interfaces needed between the different stakeholders to allow for control delegation and control sharing to happen,
- to understand how resource pooling affects the interaction between resource control and reachability mechanisms design.

5.2 Routing-based locator/identifier separation architectures

Shim6 is a host-based multihoming solution for IPv6 [Nordmark08]. It provides means for a host to handle multiple source and destination locator pairs to be used for the packets sent towards a corresponding host. It belongs to the ID/Locator split family of proposals in that the address seen by applications is not necessarily the address used by routers to forward packets. The actual mechanism that offers the mapping between identifiers and locators is a new Shim layer placed inside the IPv6 layer. In the basic definition of Shim6 the identifier is a particular case of locator, but it could also support non-routable identifiers with minimal modifications. Shim6 hosts negotiate the locators to be used between each other. This negotiation consists in exchanging the set of locators owned by each peer, so that all locator combinations can be tried in case of failure, in order to find a working alternate path through the Internet. Shim6 uses address rewriting to map upper layer identifiers to locators.

LISP is a network-based Locator/ID separation solution [Farinacci08]. It relies on an identifier to locator mapping distribution system, and on the encapsulation of packets forwarded between locator borders. In LISP, a locator is an IP Address that is routable in the Default Free Zone, and which terminates the tunnels used to carry packets from one LISP site to another. The locator to be used as the destination address for the encapsulated packet is obtained by querying the mapping distribution system for the Endpoint ID of the host to be reached. The answer will contain the locators to where packets destined to the specified ID must be encapsulated.

5.2.1 Information exposure principle.

Shim6 and LISP have been designed as purely reachability oriented proposals. Resource control has not been considered during their design and there is currently no interface planned to support the accounting, billing, or control of the resources used to make them function. The information exchange process of the Shim6 proposal is essentially host-to-host. Hence, it is hardly conceivable to let Shim6 expose information about resource control, and it would be hard to see any benefit of the knowledge by one host of the other's use of resources. On the other hand, Shim6 can be implemented to take advantage of external sources of such information to perform its locator pair selection function.

The LISP specification and implementations do not currently expose information related to resource control. LISP, however, introduces some new form of resource to be shared among the Internet routing actors, being the resource used by the mapping system to maintain its state. Mapping system updates in particular will require use of an (admittedly small) amount of resources in terms of bandwidth, storage and processing. Accounting the use of such resources by end users may be feasible, however currently the proposals for mapping systems to be used in conjunction with LISP do not integrate functionalities to control the amount of stress/variability that the system will undergo through its use by stub networks. Additionally, the use of such resources can be seen as a separate function to the LISP protocol itself.

5.2.2 Resource pooling principle

As per its ability to select the source and destination locators to be used to send the packets of a given flow, a Shim6 host pools the locator pairs used by the applications. It is up to a Shim6 host to relocate some load (at a flow granularity) over resources (locator pairs).

LISP pools the network resources (a set of locator-to-locator paths) that a Stub ISP network can use to reach an identifier. It gives the possibility to relocate resources (paths) used to reach identifiers through the use of LISP priorities and mapping system updates. LISP can spread load over resources with the weight attribute of mapping entries that define the amount of load balancing that should be done among the paths that have the same, highest, priority value.

Shim6 also supports the notion of priorities and weights. Those values can optionally be carried together with the locator sets in a locator preferences option. The use of the feature is exactly the same as for LISP, but its impact may differ due to the different location of the mechanism: LISP is active at the border of a network and thus applies policies and weights to a set of flows coming from different hosts. Shim6 is located inside end-hosts, which implies that it is up to the end host to respect or not the local policy of the network regarding traffic balance.

5.2.3 Separation of policy and mechanism principle

In Shim6, the resource pooling mechanism is instrumented through the use of Shim6 Locator preferences. The policies used to control the setting of such preferences are well decoupled from the mechanism. The choice to tweak those values is left to the implementation. More generally, Shim6 can be seen as a simple mechanism for flexibly using several paths. A host can change its current path at any time, without even notifying its peer. Each host can apply its own policy for the sending of traffic, possibly applying hints from its peer (thanks to the locator preferences option).

LISP relies on priority and weight values in the mappings for the selection of the locator pair that will support flows. The priority/weight values are the result of local policies applied by those who feed and maintain the mapping system entries. This locator pair selection mechanism is based on values whose setting is not bound to the LISP implementation. It is up to the implementation to provide an interface to apply policies on this choice. In comparison to Shim6, while the decision taker is located inside the network rather than inside the host, the choice of the path can be changed also without notifying the destination and possibly applying hints from the destination network.

5.2.4 Fuzzy end-to-end principle

Shim6 and LISP do not include application related function delegation in their scope.

However, to some extent, these proposals allow for the delegation of the selection of locator pairs to be used to carry flows.

The Shim6 specification does not mandate to allow the delegation of the locator pair selection function to other parties. However, it does not preclude this function delegation either.

In LISP, the identifier/locator mapping is undertaken in the network, at routers, away from the host – and this itself can be seen as an application of the fuzzy end-to-end principle, by delegating this function from the host to networks.

LISP Proxy Tunnel Routers [Lewis08] allow the delegation of the whole set of LISP functions from a stub network to its transit ISPs. Note however that LISP PTR primary objective is to allow interworking between LISP and non-LISP sites. Nevertheless, this mechanism could be used by ISPs that want to transfer their LISP functions (and costs) to their providers. Also, the LISP architecture currently allows entities within the mapping system to impact the priority and weight values of the mapping entries that they distribute. It is thus conceivable for a Stub Network to delegate the settings of such values to its mapping system "providers".

5.2.5 Conclusions of the analysis

What Shim6 / LISP bring to these principles compared to current state of the network?

Currently, at the routing level, load spreading can sometimes be provided through the use of IGP ECMP and BGP multipath. Some technologies, such as Cisco's Performance Routing solution [CiscoPfr], builds on multi-homed NAT boxes to provide some load balancing among sites. Hence, both Shim6 and LISP architecture proposals do bring opportunities for a wider and more flexible application of the resource pooling principle than the very limited options today. While today's pooling is only permitted through network-centric approaches, Shim6 and LISP do shift the potential for resource pooling to the host and the stub networks.

How Trilogy's work will impact these solutions regarding their respect of Trilogy's Design Principles.

By working on ISP driven path information services [Saucez08], and interfacing it with Shim6 and LISP, members of Trilogy aim at improving the respect of the resource pooling principle. It will indeed help Shim6 and LISP implementations to decide over which resources they can relocate their load. As per its definition, IDIPS allows Shim6 end systems and LISP stub networks to delegate their measurement effort to the network. From that perspective, this work will help making the end-to-end principle fuzzier.

By allowing a Shim6 implementation to take advantage of multiple locator pairs at the same time, to carry traffic between a given pair of Shim6 corresponding nodes, the work being done within Trilogy will improve the respect of the resource pooling principle, as it allows a Shim6 implementation to spread load over multiple resources.

6 Future and Related Work

The Internet architecture has lasted forty years and few real changes have been deployed in that time, despite many years of research. The difficulty of changing the Internet cannot be underestimated, so it should come as no surprise that we do not as yet have a complete proposal for a new unified Internet control architecture. Rather, we have some complementary pieces that we believe are steps towards it, and an overall framework within which they appear to co-exist. We shall continue to build these pieces into a bigger, more integrated picture, and report these efforts in publications and in our second architecture release due in 12 months time. The flip side of it being hard to change the Internet architecture is that globally, there is growing acceptance that change must happen, so the project is well-timed to impact the future Internet. We conclude this document by identifying the main areas for future work on the Trilogy architecture and how we intend to validate our architectural views, and how our work compares with other related activities.

6.1 Future directions

Below are the most critical development areas for the Trilogy architecture that we have identified, divided into the two key work areas of Design Principles and Control Architecture.

With regard to our Design Principles (Section 2), the following are the key outstanding questions and tasks.

- Information exposure: how much information is ‘sufficient’ and how much is too much? (Greater information exposure causes more coupling between modules.) We will investigate the technical consequences of different levels of exposure by evaluating strawman protocol designs, and validation in economic models is also planned.
- Resource pooling: we need to refine our understanding of what constitutes ‘effective’ resource pooling. We also intend to develop some worked examples of ensuring resource pooling mechanisms don’t conflict with each other. Finally, detailed routing mechanisms to support resource-pooling outcomes are required. We will investigate how these perform in representative Internet topologies and traffic mixes, to see what classes of solution are promising.
- We need to further refine our fuzzy end-to-end principle to define more precisely what aspects of functionality need to be distributed, and we will work on the potential function delegation protocols which cause least distortion to the basic architecture.
- Broad delegation of control was identified in our most recent output ([Trilogy08] Section 4.2.3) as a generally desirable goal, but we are currently unable to provide any solid engineering guidelines. Currently we don’t have a complete theory of the set of techniques to achieve it, but this remains an important area of activity for the project in future.

We have identified three different areas giving rise to a need to revise the Internet’s control architecture (Section 3), namely engineering problems, socio-economic problems and new control requirements. We have attempted to define an architecture that addresses each of these areas and while we believe that we have made considerable progress towards our objectives, we acknowledge further work is required to answer many specific questions. These questions are detailed throughout Section 3, but some fundamental aspects include:

- Do we have a dedicated control plane (for example, to support DDoS protection mechanisms), or is all control implicit in the data plane? To what extent is any control plane a local option, and to what extent does it have to be implemented universally to be useful?
- If there is a dedicated control plane, how does it interact with the basic reachability functions and the end-to-end control loop? How are control plane messages synchronised with the data plane operations for message routing and state installation?

- Further clarity is required with regard to our architecture extensions, and we need to better understand how we can reconcile our architectural extensions with our Design Principles. This is not straightforward as architectural purity tends to push us in one direction only. One question is therefore whether we can adapt functions to the connectionless architecture; or whether we can quantify and justify the architectural cost of the extensions.

Meaningful validation of architectural work is always a challenge. Abstract analysis can only reach a certain point: we believe that point has already been reached in the current version of the architecture. Further progress depends on evaluation of concrete case studies or new proposals. We will continue to attempt to apply the results of the economic case studies [Trilogy08] and follow-on work to the architecture presented here; in the meantime, we will develop candidate technical proposals in the areas of reachability, resource control, and their interactions. We will evaluate these both for architectural compatibility on the one hand, and simplicity and performance in realistic environments on the other. This combination of engineering evaluation, mathematical analysis, and simulation will be used to refine both the solutions and the architecture within which they fit.

6.2 Related work

We are aware of similar activities underway in all regions of the globe to re-architect the Internet. In the US, the Future Internet Design initiative [FIND] is encouraging 'blue-skies' thinking with regard to Internet architecture, while the Global Environment for Network Innovation [GENI] facility provides support in the form of a virtualised networking testbed. In Japan, the Akari [AKARI] project is also taking a clean-slate approach to the question of future Internet architecture.

The Trilogy project can be characterised as taking a more incremental approach to addressing the issues raised by the growth of the Internet. We are committed to an incrementally deployable solution that nevertheless comprehensively addresses the well-known growing pains of today's Internet. While virtualisation is not a key part of our vision, and the assumption of a clean-slate is in tension with our goal of realistic deployability, we believe that all of these activities can provide additional valuable architectural insights for our work.

Within Europe, the SmoothIT [SmoothIT] project is also addressing some of the socio-economic aspects of the future Internet, and we expect to discuss our respective approaches and results through the socio-economics sessions of the Future Internet Assembly.

6.3 Concluding Remarks

This document set out to define the overall architecture adopted by the Trilogy project as its first step towards the over-arching goal of re-architecting the Internet. We have established what we mean by the term 'architecture' and provided extensive detail of the two aspects of our architectural approach, namely our Design Principles and our holistic approach towards network and transport control. Our ambition is to develop *an architecture for change*, meaning an architecture that can adapt in a scalable, dynamic, autonomous and robust manner to local operational and business requirements. Such an architecture must avoid prejudging commercial and social outcomes for the different players so that it can reflect the outcome of contentions amongst the Internet's stakeholders: it must be 'designed for tussle'. When more fully realised, our results will significantly enhance the reliability, robustness, manageability and functionality of the Internet, and will create new and varied business opportunities based around a common control architecture.

References

[Acemoglu07] D. Acemoglu, R. Johari, and A. Ozdaglar, “Partially optimal routing. IEEE Journal of selected areas in communications”, 2007

[AKARI] “AKARI” Architecture Design Project for New Generation Network, http://akari-project.nict.go.jp/eng/index2.htm

[ALTO08] “Application-Layer Traffic Optimization (ALTO) BoF”, IETF 72, Dublin, July 2008, http://www3.ietf.org/proceedings/08jul/agenda/alto.html

[Anderson03] T. Anderson, T. Roscoe, and D. Wetherall, “Preventing Internet Denial-of-Service with Capabilities”, In Proc. HotNets II, Boston, Oct 2003.

[Andrew06] L. L. H. Andrew, K. Jacobsson, S. H. Low, M. Suchara, R. Witt, B. P. Wydrowski, “MaxNet: Theory and Implementation”, Technical Report, 2006

[ATM-PNNI96] “Private Network-Network Interface Specification”, Version 1.0, ATM Forum, March 1996

[Baker95] F. Baker (ed.), “Requirements for IP Version 4 Routers”, RFC 1812, June 1995

[Braden89] R. Braden (ed.), “Requirements for Internet Hosts - Communication Layers”, RFC 1122, October 1989

[Braden03] R. Braden, T. Faber, M. Handley, “From protocol stack to protocol heap: role-based architecture”, ACM SIGCOMM Computer Communication Review, Volume 33 , Issue 1 (January 2003)

[Briscoe05] B. Briscoe, A. Jacquet, C. Di Cairano-Gilfedder, A. Salvatori, A. Soppera and M. Koyabe, “Policing Congestion Response in an Internetwork using Re-feedback”, Proc. of ACM SIGCOMM, Philadelphia, PA, USA, Sep 2005

[Briscoe08] B. Briscoe, “Byte and Packet Congestion Notification”, Internet Draft, August 2008

[Carpenter96] B. Carpenter (ed.), “Architectural Principles of the Internet”, RFC 1958, June 1996

[Cho06] K. Cho, K. Fukuda, H. Esaki, A. Kato, “The Impact and Implications of the Growth in Residential User-to-User Traffic”, SIGCOMM 2006

[CiscoPfr] Cisco Performance Routing, *http://www.cisco.com/go/pfr*

[Clark03b] Clark, D., Braden, R., Falk, A., and Pingali, V. , “Fara: Reorganizing the Addressing Architecture”, SIGCOMM Computer Communications Review. 33, 4 (2003), 313–321.

[Clark05] D. Clark, J. Wroclawski, K. Sollins, R. Braden, “Tussle in Cyberspace: Defining Tomorrow’s Internet”, IEEE/ACM Transactions on Networking, 13(3), p. 462-475, June 2005.

[Clark88] D. Clark, “The design philosophy of the Darpa Internet protocols,” In Proc. ACM SIGCOMM, Vancouver, BC, Canada, Sept. 1988.

[Crowcroft08] J. Crowcroft and M. Bagnulo, “SNA: Sourceless Network Architecture”, Dagstuhl Seminar, June 2008

[Day07] J. Day, “Patterns in Network Architecture: A Return to Fundamentals”, Prentice Hall, December 2007

[Dukkipati05] N. Dukkipati, M. Kobayashi, R. Zhang-Shen, N. McKeown, “Processor Sharing Flows in the Internet”, IWQoS 2005

[Eardley08] P. Eardley (ed.), “Pre-Congestion Notification Architecture”, Internet Draft, August 2008

[Farinacci08] D. Farinacci, V. Fuller, D. Oran, D. Meyer, “Locator/ID Separation Protocol (LISP)”, April 2008, Internet Draft

[FIND] NSF NeTS Future Internet Design (FIND) Initiative, http://www.nets-find.net/

[Floyd00] S. Floyd, M. Handley, J. Padhye, J. Widmer, “Equation-Based Congestion Control for Unicast Applications”, SIGCOMM, August 2000

[GENI] Global Environment for Network Innovations (GENI), http://geni.net/

[Gibbens99] R. J. Gibbens, F. P. Kelly, “Resource Pricing and the Evolution of Congestion Control”, Automatica 35,1999

[Godfrey08] P. Brighten Godfrey, “Balls and bins with structure: balanced allocations on hypergraphs”, Symposium on Discrete Algorithms, San Francisco, 511-517, 2008

[Halpern08] J. M. Halpern, “Some observations on Location and Identity”, IETF 72, August 2008

[Handley03] M. Handley, S. Floyd, J. Padhye, J. Widmer, “TCP Friendly Rate Control (TFRC): Protocol Specification”, RFC 3448, January 2003

[Handley04] M. Handley and A. Greenhalgh, “Steps towards a DoS-resistant Internet Architecture”, FDNA, 2004

[Handley08] M. Handley, D. Wischik, M. Bagnulo Braun, “Multipath TCP and the Resource Pooling Principle”, TSV Area presentation, IETF, Dublin, July 2008

[Ioannidis02] J. Ioannidis and S. Bellovin, “Implementing pushback: router-based defense against DDoS attacks”, In Proc. Network and Distributed Systems Security Symposium, February 2002.

[Katabi02] D. Katabi, M. Handley, C. Rohrs, “Congestion Control for High Bandwidth-Delay Product Networks”, Sigcomm 2002

[Katti06] S. Katti, H. Rahul,W. Hu, D. Katabi, M. Médard, J. Crowcroft, “XORs in the air: practical wireless network coding”, ACM SIGCOMM Computer Communication Review, Volume 36 , Issue 4, October 2006

[Kelly98] F. P. Kelly, A. Maulloo, D. Tan, “Rate control in communication networks: shadow prices, proportional fairness and stability”, Journal of the Operational Research Society, 1998

[Kelly00] F. Kelly, “Models for a self-managed Internet”, Philosophical Transactions of the Royal Society, volume 358, 2335-2348, 2000

[Kelly05] F. Kelly , T. Voice, “Stability of end-to-end algorithms for joint routing and rate control”, ACM SIGCOMM Computer Communication Review, v.35 n.2, April 2005

[Key07] P. B. Key, L. Massoulié, D. F. Towsley, “Path Selection and Multipath Congestion Control”, INFOCOM 2007, 143-151

[Kohler06] E. Kohler, M. Handley, S. Floyd, “Designing DCCP: Congestion Control Without Reliability”, SigComm 2006

[Laws92] C.N. Laws, “Resource pooling in queueing networks with dynamic routing”, Advances in Applied Probability 24 (1992) 699-726.

[Lewis08] D. Lewis, D. Meyer, D. Farinacci, V. Fuller, “Interworking LISP with IPv4 and IPv6”, Internet Draft, July 2008

[Meyer07] D. Meyer, L. Zhang, K. Fall, (editors), “Report from the IAB Workshop on Routing and Addressing”, RFC 4984, September 2007

[Mitzenmacher01] M. Mitzenmacher, “The Power of Two Choices in Randomized Load Balancing”, IEEE Transactions on Parallel and Distributed Systems, Volume 12 , Issue 10 (October 2001)

[Nordmark08] E. Nordmark and M. Bagnulo, “Shim6 : Level 3 Multihoming Shim Protocol for IPv6”, February 2008, Internet Draft

[Pan07] R. Pan, B. Prabhakar, A. Laxmikantha, “QCN: Quantized Congestion Notification: An Overview”, May 2007

[P2PI08] Peer to Peer Infrastructure Workshop, http://www.funchords.com/p2pi/

[P4P] The P4P Working Group, http://www.openp4p.net/front/p4pwg

[Rosenberg03] J. Rosenberg, J. Weinberger, C. Huitema, R. Mahy, “STUN - Simple Traversal of User Datagram Protocol (UDP) Through Network Address Translators (NATs)”, RFC 3489, March 2003

[Roughgarden02] T. Roughgarden and E. Tardos, “How bad is selfish routing?”, Journal of the ACM, 2002

[Saltzer84] J. Saltzer, D. Reed, and D. Clark, “End-to-end arguments in system design”, ACM Transactions on Computer Systems, 2(4):277-288, Nov 1984.

[Saltzer93] J. Saltzer, “On the Naming and Binding of Network Destinations”, RFC 1498 August 1993

[Saucez08] D. Saucez, B. Donnet, L. Iannone, O. Bonaventure, “Inter-Domain Traffic Engineering in a Locator/Identifier separation context”, to appear in IEEE INM08

[Shenker95] S. Shenker, “Fundamental Design Issues for the Future Internet”, IEEE Journal on Selected Areas in Communication, 1995

[SmoothIT] Simple Economic Management Approaches of Overlay Traffic in Heterogeneous Internet Topologies (SmoothIT), http://www.smoothit.org/

[Stiemerling08] M. Stiemerling, H. Tschofenig, C. Aoun, E. Davies, “NAT/Firewall NSIS Signaling Layer Protocol (NSLP)”, Internet Draft, February, 2008

[TANA08] "Techniques for Advanced Networking Applications (TANA) BoF", IETF 72, Dublin, July 2008, http://www.ietf.org/proceedings/08jul/agenda/tana.txt

[Thaler00] D. Thaler, C. Hopps, "Multipath Issues in Unicast and Multicast Next-Hop Selection", RFC 2991, November 2000

[Trilogy08] S. Schuetz et al., "Lessons in 'Designing for Tussle' from Case Studies", Trilogy Deliverable D2, May 2008

[UPnP06] "UPnP Device Architecture 1.0", UPnP Forum, July 2006

[vanSchewick09] B. van Schewick, "Architecture and Innovation: The Role of the End-to-End Arguments in the Original Internet", forthcoming 2009, MIT Press

[Vogt07] C. Vogt, "Six/One: A Solution for Routing and Addressing in IPv6", Internet Draft, November 2007

[Yaar03] A. Yaar, A. Perrig, and D. Song. "PI: a path identification mechanism to defend against DoS attacks". In Proc. IEEE Symposium on Security and Privacy, 2003

[Yaar04] A. Yaar, A. Perrig and D. Song. "SIFF: a stateless Internet flow filter to mitigate DDoS flooding attacks". In Proc. Of the IEEE Symposium on Security and Privacy, 2004

[Yang05] X. Yang, D. Wetherall, and T. Anderson, "A DoS-limiting network architecture", In Proc. ACM Sigcomm, 2005

[Zhang08] D. Jen, D. Massey, M. Meisel, L. Wang, B. Zhang, and L. Zhang, "Routing Scalability: Separation or Elimination?", Routing Research Group, IETF 72, August 2008

Annex A Design Goal and Principle Mapping Tables

The following tables support the analysis of the Design Principles and their mapping to higher level (stakeholder) goals, as discussed in Section 4. The first table, Table 2 below, relates to the original Internet Design Principles of [Clark88] as discussed in Section 4.1.

Stakeholder Goals	**Interconnection of ARPANET with the ARPA Packet radio Network, later interconnection of academic networks**	**Robustness against Military Attack**	**Flexible support for applications, but with emphasis on remote login and file transfer.**	**Flexibility in usage of underlying networks**	**Independent Autonomous** **Networks and Endpoints**	**N/A**	**N/A**	**N/A**
Technical Design Goal	Interconnection of Existing and Future Networks	Survivability, i.e. Tolerance of an Intermediate Packet Switching Node failure	Support for Multiple Service Types, i.e. with different Bandwidth, Latency and Reliability Requirements	Support for Multiple Physical Networks, i.e. variability in the characteristics of underlying networks	Distributed Management (or Scalability)	Cost-Effectiveness	Simple Endpoint Attachment	Resource Accountability
Technical Design Decision	Interconnection by a 'Store and Forward' Packet Switched Communications Network	Essential State Information to be maintained at Endpoints rather than Intermediate Packet Switched Nodes	Separation of Network (IP) Layer from a Transport Layer able to simultaneously support different protocols that provide service differentiation e.g. UDP, TCP Endpoint-based Flow Control within TCP	Separation of Functionality between Network (IP) Layer and Transport Layer, based on a number of Design Assumptions about the Network Layer (See Note A)	Multiple Autonomous Systems, each responsible for their own self management Endpoints responsible for control of own packet flows			
Architectural Implications	Endpoints (Hosts) connected by Intermediate Packet Switching Nodes (Gateways)	Transport Layer Synchronisation Information stored in Endpoints Network Reconfiguration in response to Intermediate Packet Switched Nodes failures	'Best Effort' Datagram Service without any prioritisation, provided by Network (IP) Layer with different Transport Layer protocols (situated in the Endpoints) providing for all aspects of Service Differentiation	Responsibility for Sequenced Delivery, Broadcast/Multicast and Prioritisation of Datagrams provided at the Transport Layer rather than the Network Layer	Two-tiered routing system with Inter–AS Routing Protocol required to exchange Routing Table Information (Byte-based) Flow Control at Endpoints			
Architectural Design Principle(s)	Self-Describing Datagram	Fate Sharing	Layering E2E	Layering	Distributed Control			

Table 2 : Original Design Goals

Note: Design Assumptions about the Network (IP) Layer:

- '100 byte' Datagram Transportation with only 'reasonable' reliability.
- Basic Addressing / Routing with multipoint networks
- No Guarantee of Sequenced Delivery of Datagrams
- No Broadcast or Multicast of Datagrams
- No Prioritisation of Datagrams (see also Goal 3)
- No Support for Multiple, Service Types (see also Goal 3)
- No internal knowledge of failures, speeds or delays

The second table, Table 3 below, gives the analysis of the revised Design Principles and there relationship to the socio-economic goals of today's Internet (cf. Section 4.2).

Stakeholder (Socio-economic) Goals	**1. Global Connectivity**	**2. Business Continuity and Enhanced User Experience**	**3. Application Innovation**	**4. Network Heterogeneity**	**5. Business Autonomy and Competition**	**6. Utility Maximisation**	**7. Accessibility**	**8. Accountability**
Primary Tussles					Routing E2E Connectivity	Network Capacity	Network Access Content/Service Access	
Secondary Tussles			Content/Service Access	Network Access		Routing	Network Access	Network Capacity
Technical Design Goal	Interconnection of Existing and Future Networks	Survivability, i.e. Tolerance of an Intermediate or Edge Packet Switching Node failure	Support for Multiple Service / Application Types, i.e. with different Bandwidth, Latency and Reliability Requirements	Support for Multiple Physical Networks, i.e. variability in the characteristics of underlying networks	Distributed Management and control over E2E connectivity	Efficient Allocation of Scarce Networking Resources, including internalisation of externalities	Simple Endpoint Attachment	Resource Accountability
Technical Design Decision	Interconnection by a 'Store and Forward' Packet Switched Communications Network	Essential State Information to be maintained at Endpoints rather than Intermediate Packet Switching Node Automatic Load-balancing across multiple paths	Separation of Network (IP) Layer from Transport Layer able to simultaneously support different protocols providing service differentiation e.g. UDP, TCP	Separation of Functionality between Network (IP) Layer and Transport Layer, based on a number of Design Assumptions (See Below)	Multiple Autonomous Systems, each responsible for their own self management and policy Endpoints responsible for control of own packet flows and connectivity	Sufficient control information to be made available to support an efficient (potentially market-based) allocation of resources	Avoid undue bias in the architecture toward any one stakeholder group, in order to help promote effective market competition in the provision of Network Access and Content/ Service Access	Sufficient control information to be made available to support resource accountability
Architectural Implications	Endpoints (Hosts) connected by Intermediate Packet Switching Nodes (Gateways)	Transport Layer Synchronisation Information stored in Endpoints Multi-homing and Multi-path Network Reconfiguration in response to Intermediate Packet Switching Node failures	'Best Effort' Datagram Service without any prioritisation, provided by Network (IP) Layer with different Transport Layer protocols (in the Endpoints) providing all aspects of Service Differentiation	Sequenced Delivery, Broadcast/Multicast and Prioritisation of Datagrams all provided at the Transport Layer	Two-tiered routing system with Inter–AS Routing Protocol required to exchange Routing Table Information (Byte-based) Flow Control at Endpoints Separate policy from mechanism to allow local autonomy	Integrated mechanisms that expose information about resource usage in 'real –time'	Ease of use and configuration, with minimum architecture imposed switching costs	Extension of information exposure to allow appropriate auditing and accountability of resource usage
Architectural Design Principles	Self-Describing Datagram	Fate-Sharing Resource Pooling	Layering 'Fuzzy' E2E	Layering	Distributed Control 'Fuzzy' E2E	Information Exposure Resource Pooling	N/A (best resolved outside of these Design Principles)	Information Exposure

Table 3: Trilogy Design Goals & Principles

The rationale behind the individual columns of Table 3 is as follows.

The overarching socio-economic goal of the Internet is to provide ***global connectivity*** to all users, be they business or private users. This connectivity is still to be provided by the connectionless packet switched communications network that is the Internet, and so the most fundamental Design Principle on which the Internet Architecture is based, the Network Layer's **Self-Describing Datagram**, remains at the heart of the Trilogy Architecture. The beauty of the self-describing datagram (packet) is that no additional state is required at the intermediate nodes in order to transfer the packet to its intended destination. The only information required is included in the packet itself, from which the intermediate nodes can obtain the necessary information to correctly forward the packet. The transmission of every packet, however, makes use (albeit temporarily) of finite resources (predominantly network capacity, but also potentially state and processing resources at intermediate nodes). The Information Exposure principle can be seen as an extension of the original principle, by suggesting that the action of consuming resources should be integrated with a metric indicating the resource usage.

As the Internet increasingly becomes a critical business tool it is essential to ensure ***business continuity*** through high network reliability and availability. This has led to the adoption of multi-homing by many businesses to avoid the single point-of-failure of a connection to an individual ISP. Within Trilogy**,** we propose the principle of **Resource Pooling** as a means of automatically distributing loads across multiple paths through the network. This is an enhancement of the original Internet design in two key ways. Firstly, the use of multiple disjoint (or partially disjoint) paths provides added resilience to failure. Secondly, using multiple paths increases the overall efficiency of the network by making use of paths that would otherwise be under-utilised. Although primarily driven by business requirements, the application of resource pooling to residential users also brings the benefits of increased reliability and efficiency, and therefore an ***enhanced user experience***. In addition, Resource Pooling helps towards the achievement of the socio-economic goal of utility maximisation.

The **Layering** principle also remains a fundamental Architecture Design Principle, and is also closely related to other principles. For example, in the self-describing datagram, layering allows the intermediate forwarding nodes to only need to look at a particular layer in order to forward the packet, while leaving other layers unchanged – either for hop-by-hop modification (lower layers), or handling purely by the end systems (higher layers). We do propose, however, that the Layering principle is extended to take into account Design for Tussle. In particular, this means guarding against 'spillovers' between layers, i.e. functionality must be modularised into layers such that entities do not need to operate on multiple layers in order to process a packet.

Following on from these principles, another of the original Internet Design Principles that is still valid is that of **Fate Sharing**. This suggests that state associated with packet processing at a particular layer should only exist at nodes that do, in fact, act as endpoints for that layer. The fate-sharing principle allows routers to only process at the IP layer, which is the only information that they need to do their job, and only if a router were to fail would this knowledge require changing. Routers have no concept of 'flows', since such state is not dependent on the routers themselves – instead, this information is handled at the endpoints. Again, this principle neatly supports the 'design for tussle' goal of modularisation of function.

The principles of layering and fate sharing play a significant part in supporting the third and fourth socio-economic goals of ***application innovation*** and ***network heterogeneity***. In fact, these two goals are closely linked by recognising that the former can be thought of as the view when looking up from the network layer, the latter the view when looking down. Application innovation and diversity enhances economic welfare through the introduction of new services that provide additional benefit to end users. The achievement of this goal requires that the Internet is able to support a wide range of services and applications, with differing bandwidth, latency and reliability requirements, in a neutral and transparent manner. Conversely, network heterogeneity requires that a consistent set of services be provided to the network layer by the underlying physical networks. In many ways, this is also another example of the Layering principle: a clearly defined interface to the network layers must be presented

to the application, and in return the application must operate purely above this layer. This principle is becoming more important than ever in the rapidly evolving, diverse Internet – the architecture must not make any assumptions about the underlying network layer, be it fixed or mobile, high or low bandwidth, etc.

In order to make the Internet environment conducive to business operation and competition, and thus enhance economic welfare, it is necessary for businesses to be able to operate independently. This ***business autonomy*** will manifest itself in a number of ways, but the key point is that businesses (both end users and ISPs) can apply local policy, and operate autonomously to the maximum extent. This is facilitated by providing **Distributed Control** over Internet resources. As an example, in the case of inter-domain routing, this means allowing local policy to control how local and transit routes are presented to neighbouring domains.

An increasingly important socio-economic goal is to provide protection against misuse of the Internet, in terms of Denial of Service attacks, unwanted spam, and other behaviour that causes unnecessary inconvenience to others. At the technical level this is a tussle over E2E connectivity, and therefore the principle of Distributed Control should also be available at a finer granularity to allow end users to control who can connect to them, and who they can connect to – with connections only allowed between two parties that mutually agree to be connected. As has been discussed in Section 2, however, the 'fuzzy' E2E principle is also proposed as a way for users to delegate control of some functionality to other entities. This is seen as better representing the socio-economic needs of users, allowing delegation of features such as firewalling or spam protection.

A further socio-economic goal is that the Internet should remain a competitive marketplace, i.e. where ***business competition*** causes the market power of individual stakeholders to be restricted, whilst still providing incentives for ISPs to invest in network infrastructure, and for content/service providers to develop applications. We see it as the role of competition authorities to ensure that the marketplace for Network Access and Content/Service Access where tussles should be played out remains competitive. It is the role of architects to avoid favouring one stakeholder type over another, or one company's technological solution over another's. The architecture must be agnostic towards the application layers, while retaining a sufficiently flexible interface through which the applications can influence its behaviour. The architecture must also support competition between service providers at the network level, allowing users to switch to a provider with the most appropriate service offering, in technical and business terms.

One of the most important socio-economic goals adopted by Trilogy is that of ensuring fair usage of Internet resources. The definition of 'fair' is not straightforward but the approach we take is that of facilitating charging to be based on resource usage, rather than just flat-rate. This will promote ***utility maximisation*** through an allocation of resources based on marginal cost and willingness to pay. This goal therefore requires the introduction of market-based mechanisms (although these may lie outside the Internet Architecture, so long as some mechanism for resource usage accountability is included within the Trilogy architecture). In addition, to ensure fairness, any such mechanisms that are applied must also support the internalising of externalities. In simple terms this means that the user of a resource is charged for the external impact the consumption of that resource has on other users. The classic example is the congestion caused to other users by transmitting one's own data across the network. In order to provide support for a potentially wide range of market-based mechanisms, the principle associated with this goal therefore is limited to **Information Exposure**, i.e. the exposure of sufficient information about resource usage to support an effective and efficient allocation of that resource.

From a socio-economic perspective, ***accessibility***, i.e. ease of configuration and use, is important to ensure that access to the Internet is not exclusive to technophiles, but available to all. The benefits of ease of configuration are, however, felt by all – in particular, the easier the configuration, the easier anybody can change network provider. This naturally improves competition and thus the overall service to customers. Indeed, it is worth noting that one of the most significant motivations for NAT –

something which today is recognised as being a hindrance to protocol evolution – was a desire for ease of configuration. NATs made is much easier for customers (in the form of networks) to change ISP.

Furthermore, the architecture shouldn't force provider lock-in. An example of this is provider-assigned addressing, which increases an end user's switching costs. Allowing end users to have sufficient autonomy with their addressing (such as that provided by many locator/identifier split schemes) would greatly reduce switching costs, especially for large enterprises, and thus help competition.

The socio-economic goal of ***accountability*** is very much linked to utility maximisation by providing the assurance that resource allocation is performed efficiently and fairly. It therefore requires that the Information Exposure principle includes control information necessary to support the audibility and accountability of the allocation of scarce networking resources.