Composable, Customizable Network Services

http://www.cs.cmu.edu/~libra

Peter Steenkiste
School of Computer Science
Department of Electrical and Computer Engineering
Carnegie Mellon University

Outline

● Network services.
● Customization.
● Composition.
Motivation: Hierarchical Value-Added Services

- Programmable network infrastructure can support diverse set of services.
  - Deploy services on the fly
- Separate the development of new functionality from resources allocation.
  - Requires different areas of expertise
- Create services by building on existing components.
  - Reuse of effort
  - Specialization
  - Increased diversity

Libra Project Overview

- Build some interesting active/programmable service components.
  - Focus on distributed services, i.e. multicast, video conferencing, ..
  - Get past the programmable node, to the programmable network
- Demonstrate how richer services can be built through composition of non-trivial components.
  - Components at different levels: QoS, multicast, value-added
- Develop scalable resource selection techniques for “on the fly” service deployment.
  - Satisfy complex, competing goals
Libra Components

- Darwin delegates.
  » Application-specific QoS
- Diffserv.
  » Control over bandwidth distribution across ingress points
- End System Multicast (Narada).
  » Multicast based on overlay network between end-points
- Beagle temporal sharing.
  » Flexible temporal resource sharing across flows
- Other: communication services, distance service, video conferencing, …

- Each component is distributed in itself.
  » Rich components
- Components operate at different levels.
  » Resource .. communication .. value-added
- Programmability used for customization.
  » Change the behavior of an existing protocol

Traditional Active Networking View

- Execution environment supports the execution of active application code.
  » Language support, libraries, ..
- Active code processes packets.
  » Forwarding, processing the packet body, …
  » Active code is a mini router
- Has the flavor of an overlay network.
  » Active code does not interact with the rest of the network infrastructure
What Do We Mean By Customization?

- Base service component is extended by inserting customization code.
  - Often makes more sense than replacing the entire service implementation
  - Original thought was to replace the entire components
- Customization code changes the service behavior through API.
  - API is service specific
- Services can be very diverse.
  - Service/customization may or may not touch the data stream

Delegates: QoS Customization

- Delegates can support customizable network QoS.
- API is the router control interface.
  - Controlled access to classifier, scheduler, forwarding, monitoring
Beagle Signaling Protocol: Temporal Sharing Customization

- Temporal sharing defines how flows share resources over time.
  - Highly user specific
- General temporal sharing specifications is complex, expensive.
- TS manager supports customization.
  - Each module is quite simple
- API is very simple and inherently safe.
  - Driven by Beagle, not by extensions

Customize Bandwidth Distribution

- Customize the distribution of bandwidth across DiffServ ingress nodes dynamically.
- Meter collector collects data relevant to the user.
- Meter coordinator calculates the new distribution.
- Both are customized but only need a simple custom API.
- Prototype is based on Darwin delegates.
Some Thoughts on Customization

- Customization happened to be a convenient way to put programmability to use (for us).
  - There is typically significant overlap between existing, and new or user-specific functionality
  - As understanding improves customization can become part of the base service

- Service components can be viewed as high-function EEs with AA providing customization.
  - The EE implements a high % of the functionality!
  - Two levels of flexibility, APIs, and thrust

- Having a service-specific API is good.
  - It is much easier to define an API if you know what service you are trying to provide!
  - Implementation is driven by the API – the way it should be
  - API tends to be “narrow” – easier to make secure

Delegates: Secure QoS Customization

- Simple access control mechanism checks the validity of RCI calls.
  - ACLs set up when delegate is when delegate is set up

- Limit what flows delegates can perform operations on.
  - Based on a filter envelop

- Limit what resources delegates can manage.
  - Based on the hierarchical resource management that is used in Darwin
Outline

- Network services.
- Customization.
- Composition.

Simple Service Invocation Model

- User that needs service locates the service.
  - Can use a service location protocol, manual search, or search engine
  - Metric for selection is ad hoc
- The user then contacts and uses the service.
  - There may be some redirection under the covers
- Works well, but has drawbacks:
  - How do you use services as building blocks for richer services?
  - Restricted set of services
The Libra Service Model

Service Synthesizer

- User invokes service.
  - E.g. through web page
- Synthesizer identifies candidate service options based on user preferences and service plan.
  - Option consists of group of cooperating components
- Evaluate how each option can be realized and select best realization.
  - Both preconfigured services and on-demand services on free resources
  - Selection based on query and system optimization criteria
- Invoke service components.
  - May require recursive service invocation or service activation
- Return the data path entry points to the user.
Features of the Libra Service Model

- Separate resources from functionality.
  - Typically: first define the functionality and then select the resources
- Reuse of existing components.
  - Services are typically provided by groups of nodes
- Control over set up time versus invocation time selection.
  - Runtime functionality selection can initially be kept very simple (no AI)
- Defining the interfaces is a key issue.
  - Minimal effort in interface definition languages so far
- Programmable networking: customization of components and on-the-fly deployment components.

An Example: Video Conferencing

- In principle easy: cheap cameras, freely available software, lots of network bandwidth.
- But wait ..
- Not all video conferencing tools are compatible.
  - NetMeeting and friends are based on H.323
  - MBone tools are based on SIP
- Use of multicast is rather problematic.
  - Reasonable availability in LANs
  - But wide-area IP multicast is not very widely deployed
  - Deployment of alternatives (e.g. ESM) is very limited
Possible Video Conferencing Scenario

You must be kidding!

Possible Video Conferencing Scenario

Mbone

IP Multicast

End system multicast (ESM)

no IP Multicast

Video GW

Receive-only

NM

speaks H.323

Synthesizer Set up

User Request

Synthesizer

Service discovery

ESM provider

IPM

vic

vic

no IP Multicast

vic

vic

vic

Proxy

Receive-only

NM

speaks H.323
Execution

Summary

- Customization appears to be a simple way of putting programmability to use!
  - Alternative to reprogramming the entire service
  - Easier to make secure
- Service composition provides a way of building networks services from components.
  - Reuse existing functionality, specialization, ...
  - Separating functionality from resources can improve performance and may simplify set up
  - Automate something that is manually a very tedious process
  - Many challenges: interfaces, end-to-end properties (i.e. QOS), ...
- [Currently teaching course based on network processor evaluation system.]
- See me during break