Ubiquitous Information Access through Network-awareness and Coordinated Network Services

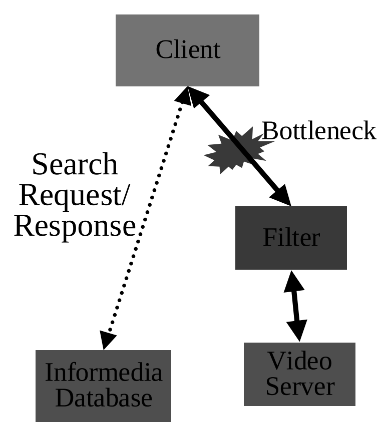
Peter Steenkiste
Computing, Media, and Communications Lab
School of Computer Science
Carnegie Mellon University

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

MM Workshop, October 1999

Adaptive Video Distribution in Informedia

- **Informedia.**
  - » **Video database with rich search capabilities**
  - » **MPEG system encoding provides good video/audio quality, and synchronization**
- **"Filter" reduces the MPEG stream to deal with bottlenecks.**
  - » **Drop B, P, I frames, in that order**
  - » **Maintain synchronization of audio and video stream**
  - » **Requires partial decoding of the MPEG system stream**
  - » **Use off the shelf software: Java Media Framework in Web browser for display**



Video Samples

- Clip 1: original MPEG video quality.
  - » Acceptable performance
- Clip 2: non-adaptive streaming.
  - » Terrible audio and video quality
  - » Clearly the wrong solution
- Clip 3: smart filtering.
  - » Performance is ok (above threshold)
  - » Occasional glitches are very annoying
  - » Would like to be able to discover bandwidth without loss of quality

Research Issues

- Infrastructure for proxy deployment.
  - » Or more general: an infrastructure for network service deployment in support of end-point applications
  - » Darwin: programmable networking
- Placement of the proxy.
  - » Or more general: resource and service selection and allocation in large scale network, trading off across resource types
  - » Remos: network information dissemination service
- Adapting to the available bandwidth.
  - » Or more general: adapting to changing conditions in the network and on the end-point (bandwidth, power, latency, ..)
  - » Darwin and Libra: network services based on coordinated components

Overview

- Motivation: Adaptive video streaming.
- Remos: an API for explicit network information retrieval.
- Darwin and Libra: high quality network services.

Examples of System-aware Applications

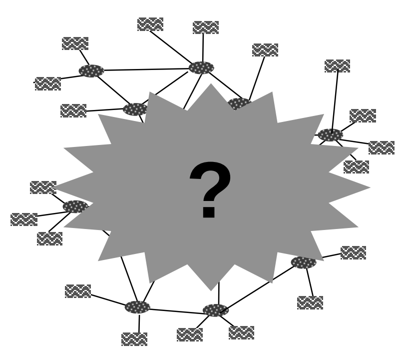
- Video streaming over the network.
  - » modify frame rate, size or resolution to match bandwidth
- Image databases.
  - » apply different image resolutions or compression algorithms
- Access to remote storage servers.
  - » select fastest server (covers network plus server)
- Visualization of large data sets.
  - » Where is processing and rendering performed?
- Common theme: *application-specific* adaptation.
  - » Protocol such as TCP/IP provide generic adaptation
- Adaptation can be lossless or lossy.
  - » Change resources that are used versus reduction in timeliness, accuracy, or precision

System and Network-aware Applications

- Adapt to static network characteristics.
  - » Shared Ethernet versus switched gigabit ethernet
  - » Pick the right configuration at startup
  - » Easier since the application is stateless
- Adapt to dynamic network conditions.
  - » Changes in load conditions
  - » Can be done periodically or continuously
  - » Harder - application has to be explicitly designed for it
- Need information about the network.
  - » Capabilities of components, topology and location information, ..
  - » Dynamic resource availability

Network-Application Coupling

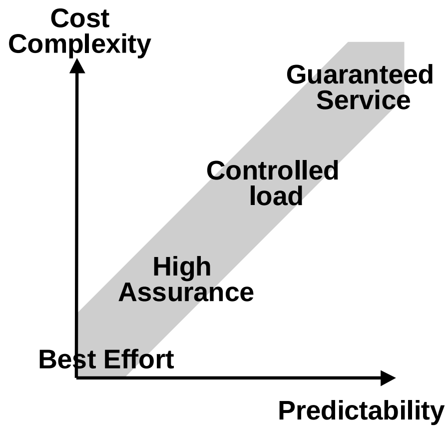
- Implicit feedback.
  - » Application deduces network status information from observed performance
    - – e.g. TCP, video streaming
  - » easy to support; hard to use
  - » very portable (but assumptions!)
  - » very common today
- Explicit feedback.
  - » Network provides explicit information about conditions
  - » e.g. ATM ABR
  - » harder to support; easier to use
  - » often network specific
  - » only rarely supported



➨ what type of feedback should be supported?

Integrated Services Networks: Good News and Bad News

- Good news: can pick service class that best matches application needs.
  - » e.g., guaranteed service for video
- Bad news: more choices and complexity!
  - » must choose the service class
  - » must adapt to service availability
  - » classes are only being deployed slowly; heterogeneity
- Cost issue: you pay for predictability.
  - » pressure to work well with the more basic service classes



➨ how do applications make effective choices?

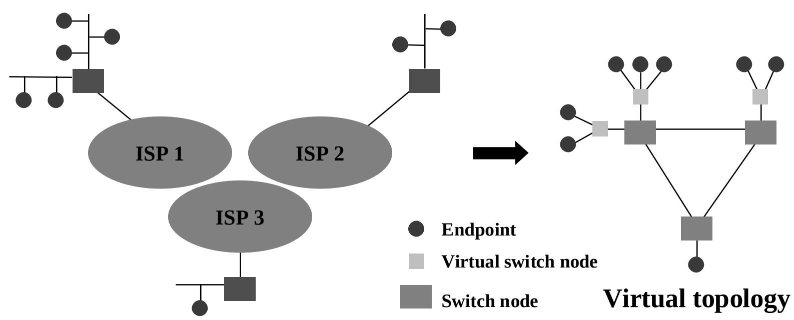
Need Range of Solutions

- Reservations for small set of hard-real time applications or to maintain minimal QoS.
  - » Need spectrum of QoS services: hard reservations, controlled resource sharing, best effort
  - » Key concerns: performance and scalability
  - » Focus of the Darwin project
- Make services and applications adaptive so they can perform well in a wide range of environments.
  - » Need adaptation frameworks that support diverse application-specific adaptation paradigms
  - » Need system information to drive adaptation
  - » Focus of Remos

Remos

- API that allows applications to collect explicit information on network connectivity.
  - » considers sharing, simultaneous queries
- Topology-based queries allow applications to explore a large space of options.
  - » use of “virtual” topology simplifies interface
- Flow-based queries provide feedback on performance for specific traffic patterns.
  - » flows can have different types
- Goal is to capture a broad set of information.
  - » bandwidth, latency
  - » static and dynamic information
  - » variability and accuracy

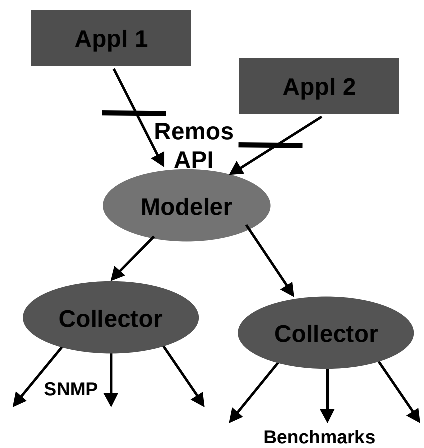
Virtual Topology



- Virtual topology summarizes connectivity.
  - » links annotated with bandwidth or latency information
- More concise than pair-wise connectivity information.
- Simpler than actual topology.

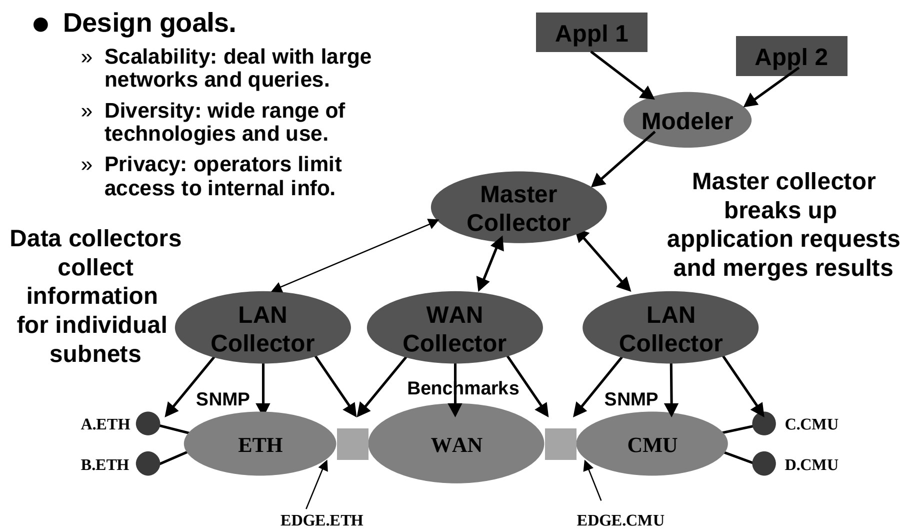
Implementation

- **Collector collects raw network information.**
  - » prune the information space
  - » collect static and dynamic information
  - » periodically refresh information
  - » uses caching to improve efficiency and response time
- **Modeler uses raw information to answer specific queries.**
- **Can insert predictors into the architecture.**
  - » Estimate future conditions using models of past usage
  - » Place between modeler and collectors



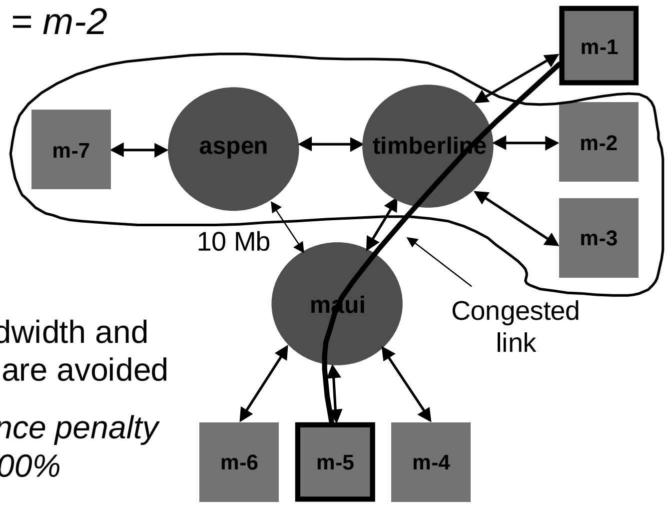
Scalable Remos Architecture

- **Design goals.**
  - » Scalability: deal with large networks and queries.
  - » Diversity: wide range of technologies and use.
  - » Privacy: operators limit access to internal info.



Finding Set of Nodes with Best Connectivity

start node = m-2

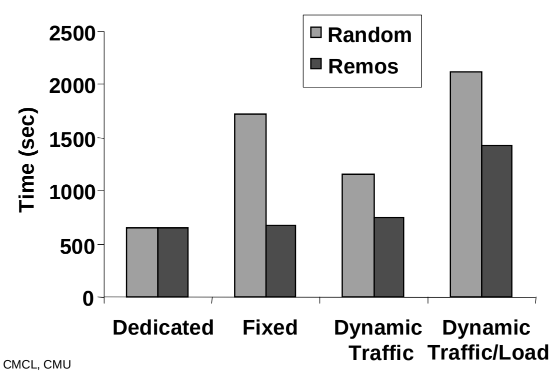


Low bandwidth and busy links are avoided

Performance penalty ~200%

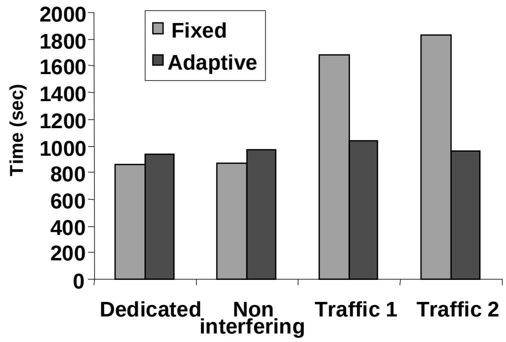
Startup Adaptation in Airshed

- **Automatically avoid busy links and nodes.**
  - » **Requires explicit information**
- **Helps even under high loads.**



Runtime Adaptation in Airshed

- Dynamic adaptation after major time steps.
  - » Complexity of adaptation dealt with by Fx runtime system
- Improves efficiency under a wider range of conditions
  - » But adds overhead (querying Remos about network conditions, adaptation inefficiencies)

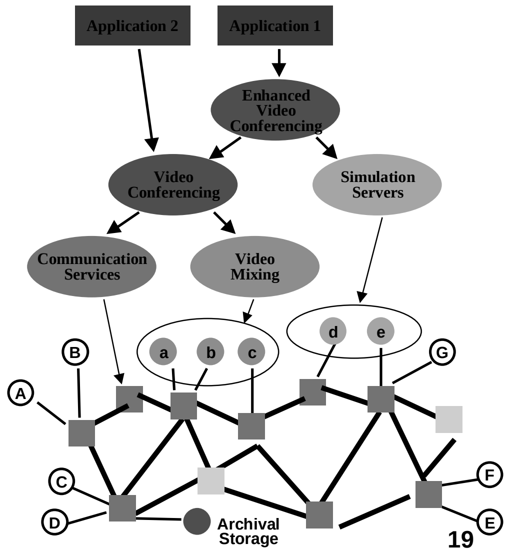


Discussion

- Access to system status information can help thin clients deal with scarce resources.
  - » Probably becomes more important as client is thinner
- Extending the scope of explicit information can help clients use resources more efficiently.
  - » What network interface should I use?
  - » Are reservations available? Are they needed?
- Explicit information makes it possible to consider a large space of options.
  - » Probing becomes impractical after a while
- Many open research issues.
  - » Cost of information gathering, required accuracy, ...

Hierarchical Services

- Services organized in a hierarchy.
  - » complex services built on primitive services
  - » applications at the top
  - » supports specialization
- Clients can be thin or thick.
- Users want predictable high quality services, but the infrastructure is unreliable and shared.
- Services must meet user and application-specific QoS criteria, but the infrastructure is generic.

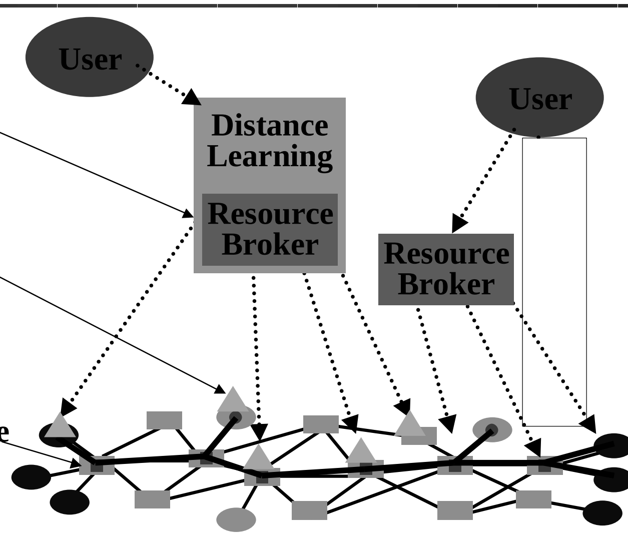


An Architecture for Application-level QoS

Resource Brokers perform global resource allocation on behalf of the user

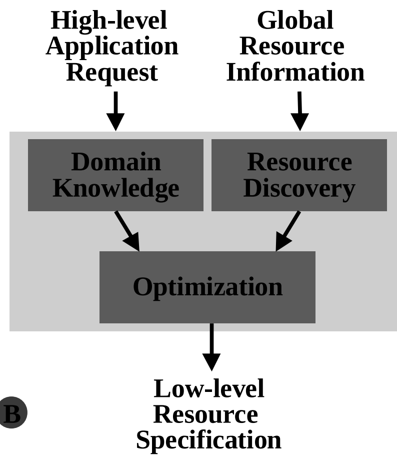
User agents manage the user's network resources at runtime

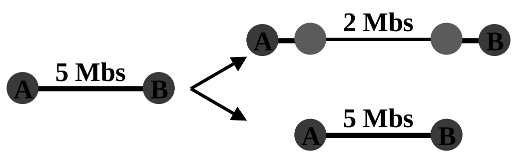
Schedulers guarantee low-level properties for traffic flows



Resource Brokers Perform Global Resource Allocation

- Map high-level user request onto a set of resources using domain knowledge.
  - » Request includes cost constraints and QoS metrics
  - » E.g. 1.5 Mbs for MPEG-1 video
  - » Generate alternate mappings, e.g. use compression or not
- Optimize QoS or cost.
  - » Select "the best" mapping option
  - » Tradeoffs across resource types
  - » Negotiate with user about options



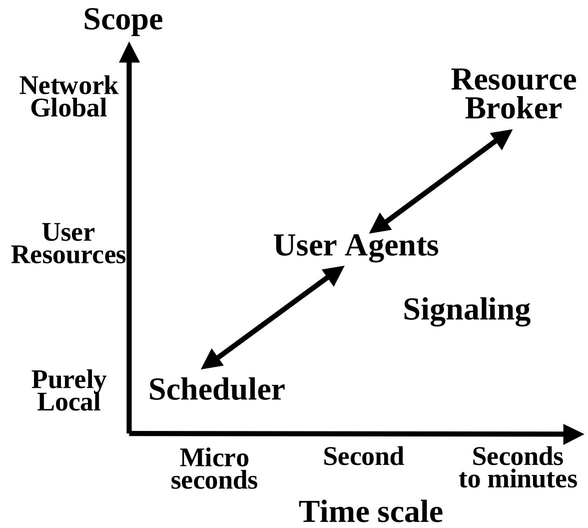


Delegates Manage Resources at Runtime on Behalf of the User

- Motivation: service quality is sensitive to how resources are managed at runtime.
- Code insertion is a very flexible method of adding "domain expertise" to the network.
  - » Set of parameters is too restrictive and too static
  - » "Applets in reverse": user extends network functionality
- Executing delegates on the router results in fast response times.
  - » Immediate access to relevant information
  - » Eliminate roundtrip delay to endpoints
  - » Active networking in the control path

Why Not a Single Mechanism?

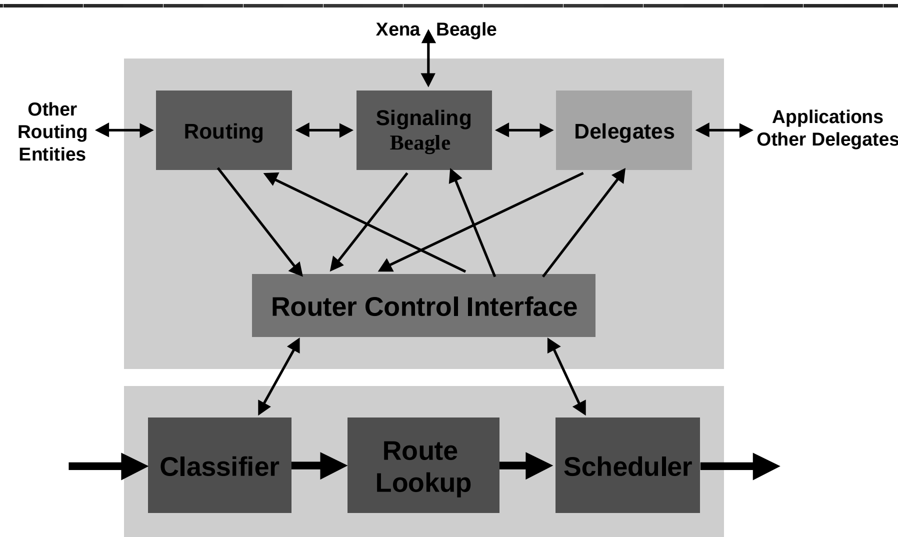
- Propagation delays and volume of information limit the speed of global actions.
  - » E.g. information collection, control
- Need multiple mechanisms with different scope and time scale.
  - » Fast local actions: deal quickly with dynamic aspect of QoS
  - » Slower global actions: exploit domain knowledge



Peter A. Steenkiste, CMCL, CMU

23

Router Architecture

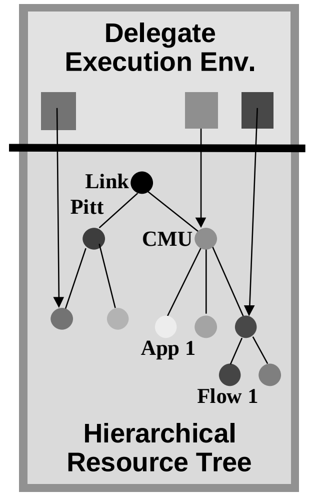


Peter A. Steenkiste, CMCL, CMU

24

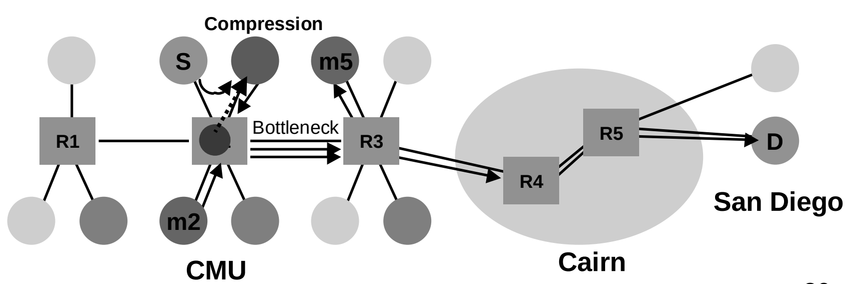
Delegate Implementation

- Delegates implemented as Java code segments.
  - » Sandboxing provides basic safety
- RCI implemented as native class
  - » Monitoring: bandwidth, queue size, ..
  - » Resource management: modify scheduler parameters, ..
  - » Flow redirection: tunneling, ..
  - » Delegate communication
- Clean programming model.
  - » Delegates are associated with nodes in the resource tree
  - » Scheduler: isolation in data plane
  - » Delegates: isolation of control

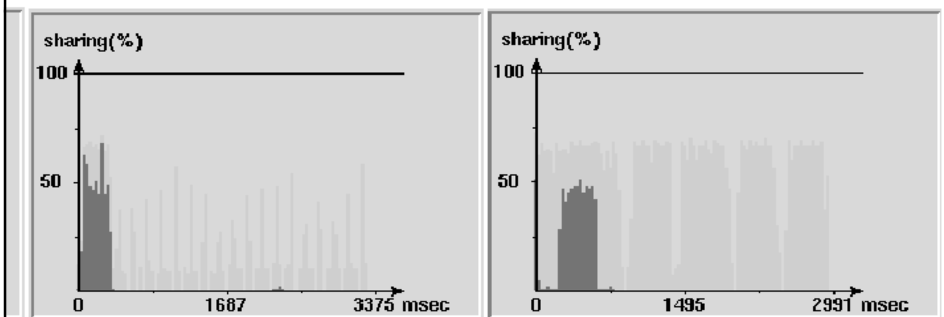


Dynamic Control of Compression of Motion JPEG Stream

- Control delegate controls the compression level of video compression engine based on router status.
  - » JPEG hardware encoder in this example
- The delegate gives the user more information and the information is available quicker.
  - » The alternative is to wait for the receiver to give feedback



Fast Response Improves Video Quality



No delegate:
Separate resource pools
Low video quality

With delegate:
Shared reservation
Video opportunistically uses free data bandwidth

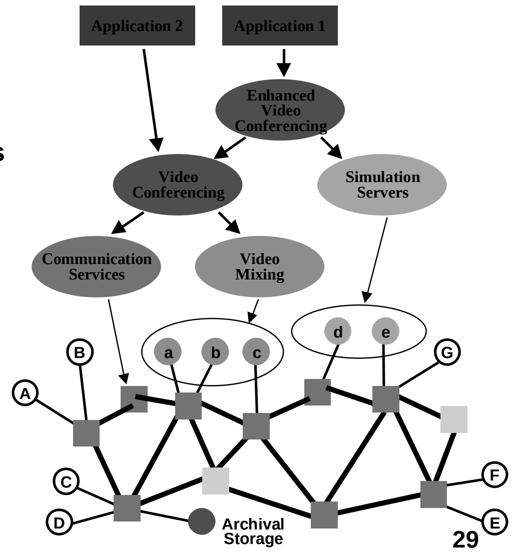
Discussion

- **Network services are the right way to go for thin <u>and</u> thick clients.**
  - » **Nobody wants to manage a computer system**
  - » **Issue is more pressing for the thin clients, of course**
- **Unfortunately, relying on network services is much harder for mobile clients.**
  - » **The last hop is the weak link, I.e. the clients should be able to work fairly independently**
  - » **Battery power may limit use of network**
  - » **Argues that mobile clients should be thicker, which is of course counter intuitive**
- **Many open research issues.**
  - » **Access to infrastructure, standard runtime environments, mobility, billing and accounting, ..**

Scalable Advanced Network Services

Service = functionality + resources

- Programmability of network resources allows separation of concerns.
  - » Identify functions and then locate resources
  - » More flexibility
  - » More opportunity for optimization
- How to provide service functionality using the Darwin resource infrastructure as a foundation?



Electronic Services: Challenges

- Distributed versus localized services.
  - » Localized: single location, e.g. a Web server
  - » Distributed: requires set of nodes to work together, e.g. Web servers plus coordinated caches
  - » Distributed services require coordination, e.g. protocols
- Composable versus monolithic.
  - » Monolithic: self-contained
  - » Composable: builds on lower level services
  - » supports reuse of functionality and specialization
  - » different from traditional software composition (distributed) or distributed computing (location matters)
- User end-to-end properties associated with the total service.
  - » Challenging for composite distributed services

Libra: Scalable Advanced Services based on Coordinated Components

- Core networking services.
  - » Active core-stateless QoS
- Middleware services.
  - » Support for end-to-end non-functional requirements
  - » Load balancing based on request contents
  - » Active distributed monitoring
- User-level services.
  - » Virtual private network service with QoS support
  - » Proxy runtime environment for mobile hosts
  - » Scalable video distribution
- Choice of services emphasizes distribution, composability, and end-to-end requirements.
  - » Based on an extended version of Darwin's resource mesh

Remos and Darwin Teams

- Peter Dinda.
- Thomas Gross.
- An-Cheng Huang.
- Peter Lieu.
- Bruce Lowekamp.
- Nancy Miller.
- Dave O'Hallaron.
- Peter Steenkiste.
- Jaspal Subhlok.
- Dean Sutherland.
- Prashant Chandra.
- Yang-Hua Chu.
- Allan Fisher.
- Jun Gao.
- Corey Kosak.
- Eugene Ng.
- Umair Shah.
- Ion Stoica.
- Eduardo Takahashi.
- Hui Zhang.