Distributed Adaptation for Heterogeneous Networks

Mark Yarvis

July 27, 2000
Roadmap

» Adaptation and network heterogeneity
  • Our approach: distributed adaptation
  • Advantages of distributed adaptation
  • Conductor: design and implementation
    – Architecture
    – Stream Management
    – Reliability
    – Planning
    – Security
The Need for Adaptability

• Networks: not always fast and free
  – Bandwidth, latency, jitter, $$, security, reliability

• Applications typically assume a minimum level of network service
  – Cost vs. benefit imbalance

• Goal: applications should provide gracefully degraded service
**Adaptive Software:**

Software that can tailor its services to constraints in available resources and user expectations.
Enabling Adaptability

• Adapt application-layer protocols from within the network
  – Compress, encrypt, prefetch
  – Distill a video stream to black-and-white
  – Remove advertisements from web pages
  – Prioritize interactive browsing over downloads
  – Power down wireless interface during predicted query response latency

• Is this heresy?
Trend: Network Heterogeneity

- "Last mile"
- Adapt here
- Backbone
Trend: Network Heterogeneity

Internet

768Kb/s

Connectors: Telephone, TV, and Internet Access

People: Working, Relaxing, and Traveling

Technology: Computers, Data Centers, and Network Nodes
Adaptation in Heterogeneous Networks

- Multiple constrained links
- Multiple types of constraints
- Conditions difficult to predict
- Many possible adaptations
- Many possible locations for adaptation
Roadmap

• Adaptation and network heterogeneity
  » Our approach: distributed adaptation
• Advantages of distributed adaptation
• Conductor: design and implementation
  – Architecture
  – Stream Management
  – Reliability
  – Planning
  – Security
Distributed Adaptation

- **Goal:** allow applications to degrade gracefully in heterogeneous networks
- **Required:**
  - Multiple adaptations
  - Distributed within the network
  - Coordinated
The Conductor Approach

• Arbitrary (and potentially lossy) adaptation of application-level protocols
  – Reliable connection-oriented streams

• Dynamic selection of adaptive code modules at enabled points in the network
  – Conductor is incrementally deployable

• Application transparent, but not user transparent
  – User controllable
Challenges Met by Conductor

• New reliability model required
  – Exactly-once delivery of bytes no longer makes sense

• Enable coordinated adaptation
  – Multi-node planning in a low-performance network

• Security without de facto infrastructure
  – Protect control over adaptation without a ubiquitous authentication architecture
Roadmap

• Adaptation and network heterogeneity
• Our approach: distributed adaptation
  » Advantages of distributed adaptation
• Conductor: design and implementation
  – Architecture
  – Stream Management
  – Reliability
  – Planning
  – Security
Case Study #1

Secure, Low-Bandwidth Web Browsing
Case Study #1

Requires stream access!

Prioritize

Shortest job first?
Text before images?
Case Study #1

Insecure  Low Bandwidth  Insecure

Encrypt  Prioritize  No data access

WaveLAN

768Kb/s

Internet

B of A

Web Server
Case Study #1

WaveLAN

Insecure

Low Bandwidth

Insecure

Encrypt

Encrypt

Prioritize

Internet

B of A

Web Server

768Kb/s
Case Study #2

Wireless to Wireless Video Streaming
Case Study #2
Case Study #2

Ricochet 2
128Kb/s

Internet

WaveLAN
6Mb/s

Lower BW

Low Bandwidth

Drop Frames

Compress
Case Study #2

Ricochet 2
128Kb/s

Internet

WaveLAN
6Mb/s

Lower BW

Low Bandwidth

Drop Frames
Case Study #2

WaveLAN 6Mb/s

Low Bandwidth

Cache

Internet

Ricochet 2 128Kb/s

Lower BW

Compress

Drop Frames
Case Study Results

- Multiple adaptations
- Multiple points of adaptation
- Coordination required!!!
- Must understand end-to-end network characteristics
Adaptation Deployment Constraints

- Limited node resources
  - Load balancing, palmtops
- Location, location, location
  - Proximity means agility
  - Hardware access
  - Leveraging topology
- Conflicting adaptations
Requires user diagnosis and intervention «
May require specialized applications «

- Text-based web browsers
- Palm clipping apps

Specific applications

Other Approaches
Other Approaches

• Adaptable applications
  - Odyssey [Noble]
  - Rover [Joseph]
  - Application partitioning [Kottmann][Watson]

  » Requires application modifications
  » Application writer must foresee and understand possible network conditions
Other Approaches

• Adaptation as a network service
  – Boosting existing protocols
    • Snoop [Balakrishnan], Protocol Boosters [Mallet]
  – Protocol Transformers
    • Transformer Tunnels [Sudame, Badrinath]
    • Proxy architectures [Fox, Gribble] [Zenel]
  – Active Networks
    » Lack coordination and reliability needed for arbitrary multipoint adaptation
Roadmap

- Adaptation and network heterogeneity
- Our approach: distributed adaptation
- Advantages of distributed adaptation
  » Conductor: design and implementation
    - Architecture
    - Stream Management
    - Reliability
    - Planning
    - Security
Conductor Architecture

• Components: framework and adaptation modules

• Adaptation framework
  – Transparent interception and routing
  – Node/link status monitoring
  – Distributed planning and deployment
  – Adaptor runtime environment
Conductor Architecture

• Adaptor modules
  – Operate on data stream
    • Arbitrary modifications allowed
  – Easily extensible set
  – Frequently paired
  – Composable
  – Stored on Conductor-enabled nodes
Adaptor Deployment

Split-TCP

Unpaired adaptor

Conductor framework

Adaptor pair

Conductor-enabled nodes
A Conductor-Enabled Node
Stream Management

• Capture at socket level
  – Maintain existing socket API
  – Route through other Conductor nodes
  – Create transparent split-TCP connection

• Stream identification
  – Port numbers
  – Protocol identifier
  – Magic number
Reliable Transmission

- **Goal**: Provide adaptation for applications that expect reliable delivery
  - TCP, exactly-once delivery of bytes

- **Adaptation can violate typical assumption of data immutability**
  - Must allow intentional data loss
  - Exactly-once delivery of transmitted bytes makes no sense
Reliability and Adaptation

• Possible failures: adaptors, nodes, links
• Failure modes
  – Potential data loss
  – Partial adaptation of data
  – Lost adaptor state
  – Adaptor consistency
Reliability in Conductor

• End-to-end connection built using multi-split-TCP
  – Reliability between points of adaptation
  – Leverage existing technology
  – Adaptation at each node independent of TCP

• Node and link failures detected as TCP connection failures
Reliability in Conductor

• How do we know if any data was lost?
• From what point should transmission be restarted?

  » Need a new unit of retransmission
  » Maintain some correlation between pre- and post-adapted data
Reliability in Conductor

- **Semantic Segmentation**: a semantically meaningful unit of retransmission
  - Divide stream into semantic units
    - Dynamically, based on data type and adaptation
    - No application hints required
  - Preserve semantic meaning of each segment end-to-end
    - Maintained by segment combination
  - Adaptors can express recovery constraints
Rules of Segmentation

• Start with one byte segments
• Constrain each stream modification to one segment
• Combine segments where necessary
  • Not reversible
  • New segment contains combined semantic meaning
• Final delivery of complete segments only
Reversing Segmentation

• With lossy adaptation, segments must remain until delivery
  » Must handle this case
  
  <img src=a.gif> ➔ <img lows<a <src=b.gif ...> ➔ ?

• Lossless adaptation potentially allows original segmentation to be restored
  » A possible optimization

[Diagram of filmstrips showing lossless adaptation process]
Benefits of Segmentation

• Service guarantees:
  – Transaction-like adaptation (all or nothing)
  – Exactly-once delivery of some form of each semantic element

• Adaptors can express appropriate points for adaptation changes
Adaptor Selection

• Goal: Select an appropriate set of adaptors for end-to-end conditions
  – Requires a planning capability

• Issues:
  – Speed
    • Planning must occur before data flows
  – Cost
    • Likely presence of low-quality links
  – Coordination
    • Local decisions are not always best
Adaptor Selection

• Inputs to “plan formulation”
  – Node characteristics
    • Resources: CPU, disk, available adaptors
    • Security constraints
  – Link characteristics
    • Bandwidth, latency, etc.
    • Current, historical, expected
  – Data Characteristics
  – User preferences
    • Important data qualities and costs
Planning in Conductor

• Centralized planning
  – Gather all inputs to one location
  – Formulate plan
    • Pluggable architecture
  – Distribute plan

• Reaction to changing conditions
  – Adaptors handle a range of conditions
  – When tolerances are exceeded, replanning occurs
Planning in Conductor

Node 1

Node 2

Node 3

Node 4

Node 1 Info

Nodes 1,2 Info

Nodes 1,2,3 Info

Plan

Plan

Plan

Data flow begins

Formulate Plan
Planning in Conductor

• Benefits:
  – Only requires one round trip latency
  – Can plug in any “plan formulation” code
    • Static
    • Template based
    • Heuristic search based
Securing Distributed Adaptation

- **Goals:**
  - Maintain endpoint control over adaptor selection and deployment
  - Protect user data

- **Key difficulties**
  - Cross-domain node participation
  - No ubiquitous authentication mechanism
  - Varying user requirements
Security in Conductor

• Solutions:
  – Security monitor controls planning messages
    • Messages can be authenticated
    • Dynamically pluggable authentication scheme
      – Selected at an endpoint
    • How do we ensure everyone uses the same authentication scheme?
  – Encryption adaptors protect user data
    • Still need secure key distribution
Security in Conductor

Node 1
- Auth. Scheme
- Node 1 Info (Node 1)

Node 2
- Auth. Scheme
- Node 1 Info (Node 1)
- Node 2 Info (Node 2)

Node 3
- Auth. Scheme
- Node 1 Info (Node 1)
- Node 2 Info (Node 2)
- Node 3 Info (Node 3)

Node 4
- Auth. Scheme
- Node 1 Info (Node 1)
- Node 2 Info (Node 2)
- Node 3 Info (Node 3)
- Node 4 Info (Node 4)

Authenticate
Formulate Plan

Authenticate and verify

Plan
Node 4

Authenticate
Security in Conductor

• Authentication schemes
  – None
  – Public key encryption
    • Hierarchical key service
    • Chain of trust
  – Kerberos

• Key distribution
  – Based on authentication scheme
Implementation Status

• Stream management
  – Interception based on port number
  – Routing based on underlying routing

• Reliability
  – Semantic segmentation: implemented
    • Adaptor API
  – Recovery protocol: partially implemented
Implementation Status

- Planning
  - Information gathering protocol: implemented
  - Simple planner and environment monitor

- Security
  - Security architecture: implemented
  - Several authentication mechanisms
  - Sample encryption adaptors: implemented
Implementation Status

- Completing the implementation
  - Suite of useful adaptors
  - Dynamic “plan formulation” algorithm
  - Complete implementation of the recovery algorithm
Measurement of Success

• Effectiveness
  – Construct examples similar to case studies

• Low overhead
  – Measure overheads when adaptation is not required

• Complete services
  – Dynamic demo: automatically deploy, respond to drastic changes, cope with failure
Measurement of Success

• Usability
  – Everyday use in a heterogeneous office environment
# Schedule

<table>
<thead>
<tr>
<th>Month</th>
<th>Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sep</td>
<td>Initial Office Deployment</td>
</tr>
<tr>
<td>Oct</td>
<td>» Adaptor suite</td>
</tr>
<tr>
<td>Nov</td>
<td>Development: » Dynamic planning</td>
</tr>
<tr>
<td>Dec</td>
<td>» Recovery protocol</td>
</tr>
<tr>
<td>Jan</td>
<td>Dynamic Demo</td>
</tr>
<tr>
<td>Feb</td>
<td>Measurements</td>
</tr>
<tr>
<td>Mar</td>
<td></td>
</tr>
<tr>
<td>Apr</td>
<td>Dissertation</td>
</tr>
<tr>
<td>May</td>
<td></td>
</tr>
</tbody>
</table>
Contributions of This Work

• Design: architecture to make distributed adaptation possible

• Technical: new model and algorithms for reliability in the face of adaptation
  – Semantic Segmentation

• Engineering: a deployable system

• Demonstration: fully application-unaware adaptation is feasible
Conclusions

• In heterogeneous networks **distributed adaptation** enables graceful degradation

• **Conductor** enables distributed adaptation
  - First design and implementation of distributed adaptation
  - Reliability model compatible with adaptation
  - Architecture for coordinated adaptation
  - Trusted coordination for disjoint nodes