Conductor: Distributed Adaptation for Heterogeneous Networks

Mark Yarvis
yarvis@cs.ucla.edu

http://fmg.cs.ucla.edu/Conductor

November 8, 2001
Introduction

- **Problem:** Application behave poorly in highly variable and heterogeneous environments
- **Goal:** Help applications provide the best possible service to the user given current network conditions
- **Approach:** Conductor provides coordinated and distributed adaptation of application-level protocols as a transparent middleware service
The Need for Adaptability

- Networks can be highly variable
  - Bandwidth, latency, jitter, $$, security, reliability

- Applications frequently assume a minimum level of network service
  - Cost vs. benefit imbalance

- Applications should provide a level of service that the network can support
Enabling Adaptability

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

"Last mile"  Adapt here  Backbone

56Kb/s
Trend: Network Heterogeneity

WaveLAN

768Kb/s

Introduction — Conductor — Selected Details — Results
Distributed Adaptation

• Goal: Help applications provide the best possible service to the user given current network conditions

• Required:
  – Multiple adaptations
  – Distributed within the network
  – Coordinated
Case Study #1

Secure, Low-Bandwidth Web Browsing
Case Study #1

Requires stream access!

56Kb/s

Low Bandwidth

Prioritize

Shortest job first?
Text before images?

Introduction — Conductor — Selected Details — Results
Case Study #1

Introduction — Conductor — Selected Details — Results
Case Study #2

Wireless to Wireless Video Streaming
Case Study #2

User A
WaveLAN 2 Mb/s
Low Bandwidth

Internet

User B
WaveLAN 2 Mb/s
Low Bandwidth

Compress

Introduction — Conductor — Selected Details — Results
Case Study #2

User A
Minstrel
19.2 Kb/s
Lower Bandwidth
Drop Frames

Internet

User B
WaveLAN
2 Mb/s
Low Bandwidth
Compress

Introduction — Conductor — Selected Details — Results
Deployment Constraints

• Limited node resources
  – Load balancing, palmtops
• Location, location, location
  – Proximity means agility
  – Hardware access
  – Leveraging topology
• Conflicting adaptations
Adaptation in Heterogeneous Networks

• Must consider end-to-end network characteristics
  – Multiple constrained links
  – Multiple types of constraints
  – Conditions difficult to predict
• Many possible adaptations
• Multiple points of adaptation
• Coordination required!
Conductor: Architecture Overview ...

- Our Approach
- Conductor’s Architecture
- Stream Management
- Adaptor Selection
- Security
- Reliability
- Adaptation-aware API
The Conductor Approach

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

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

• Application transparent, but not user transparent
  – User controllable
Conductor Architecture

- Components: framework and adaptation modules
- Adaptation framework
  - Transparent interception and routing
  - Node/link status monitoring
  - Centralized 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
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
  - Dynamic, fine-grained identification by adaptors
Adaptor Selection

- **Goal:** Automatically select appropriate sets of adaptors for end-to-end conditions
- **Issues:**
  - Speed, cost, coordination
- **Plan based on distributed information**
  - Node and link characteristics
  - Data characteristics
  - User preferences
  - Available adaptors
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
The Planning Protocol

Node A | Node B | Node C | Node D

Formulate Plan

Node Information | The Plan | User Data
What should be protected?

• Protect the nodes from misbehaving adaptors
  – Leverage existing research

• Protect the user from misbehaving nodes
  – Allow only desired adaptations

• Protect the secrecy and integrity of the user data
  – But, still allow adaptation
Security in Conductor

• Protect planning from untrusted nodes
  – Implicitly trust endpoints
  – Authenticate other nodes and establish trust
• Problem: no ubiquitous authentication mechanism
  – Conductor allows dynamic selection and enforcement of an authentication scheme
• Adapt plaintext only at trusted nodes
  – Encrypt user data between trusted nodes
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 in Conductor

• Possible failures: nodes, links, adaptors
• New reliability model
  – Exactly-once delivery of semantic elements
• Semantic segmentation
  – Dynamic and automatic stream checkpointing
  – Ensures that adaptation is atomic
  – Provides exactly-once, in order delivery of the adapted stream
Reliability in Conductor

• Recovering from adaptor failure
  – Identify lost adaptors
    • Maintain distributed state describing adaptor pairing and composition
  – Restore adaptor consistency
    • Adaptor state is lost
    • Cannot just replace failed adaptor, in the general case
    • Remove paired and composed adaptors
  – Replan and redeploy as required
Adaptation Aware Apps

- Conductor provides transparency through automatic services:
  - Interception, planning, reliability, adaptation
- But application knowledge can be useful
- An API can give some apps more control
  - Select and control adaptors
  - Select trusted nodes
  - Provide data for retransmission
- The best of both worlds
Evaluating Conductor

- Effective delivery of adaptation
  - Significant benefit in three case studies
  - Low overheads
  - Demonstration of failure recovery
- Office deployment
  - Daily use for POP3 protocol
- A platform for distributed adaptation
  - Beta software release
    - http://fmg.cs.ucla.edu/Conductor
  - A basis for further research
In Greater Detail …

- Conductor Reliability
- Conductor Security
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
<img src=a.gif>  

< img  low=a.gif >  

www Client  

src=b.gif src=a.gif  

WWW Server  

Introduction — Conductor — **Selected Details** — Results
Reliability in Conductor

• How do we know if any data was lost?
• Was adaptation complete?
• 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
  • New segment contains combined semantic meaning
  • Assign segment ID from last combined segment
• Final delivery of complete segments only
Benefits of Segmentation

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

• Adaptors can express appropriate points for adaptation changes
Threats to Adaptor Selection

The network is dreadfully slow and insecure!

Deploy these adaptors!

The network is fast and secure!

B of A

Web Server

Internet

768Kb/s

Insecure

Low Bandwidth

WaveLAN

Introduction — Conductor — Selected Details — Results
What nodes can we trust?

- Various levels of trust possible
  - See or modify plain text
  - See or modify encrypted text
  - None
- Implicitly trust endpoints
- Trusting other nodes
  - Requires some type of authentication
  - Static list, dynamic trust model
Complications of Distributed Adaptation

- Users require different levels of security
- Adaptation may span administrative domains
  - No ubiquitous authentication infrastructure
  - Many choices; how do we agree securely?
- Must allow limited stream access within the network
  - Only desired adaptations
  - Typically restricted to trusted nodes
Authentication

• Goals:
  – Verifiable node identity
  – Digital signature capability
• Plug-in modules provide various authentication schemes
  – Null
  – Public-key based: tree, chain of trust
  – Kerberos based
Secure Planning

• Self-enforcing scheme selection
  – The client selects an authentication scheme
  – The server returns a signed message indicating the scheme used
• Authentication
  – Each node authenticates to the planner
  – The planner authenticates to each node
• Secure planning
  – Planning information is signed by the sender
  – Use only authentic information from trusted nodes
  – The plan is signed by the planner
Virtual Link Encryption

• Allow plaintext adaptation only at trusted nodes
• Encrypt between points of adaptation
  – Use encryption adaptors
• Requires:
  – Selection of trusted nodes
  – Encryption adaptor selection and deployment
  – Secure key distribution
Research Results ...

- Performance
- Comparison with other research
- Key contributions
- Conclusions
Selected Performance Results

- Overheads reduce the potential benefit of adaptation
  - Conductor has low startup and data handling costs
- The framework is only useful if adaptors can provide real benefit
  - Conductor provided significant benefit in our case studies
Conductor Overheads

• Data handling overheads
  – Reduction of throughput and latency over 100 Mbps Ethernet

<table>
<thead>
<tr>
<th></th>
<th>Per enabled node</th>
<th>Per null adaptor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Throughput Reduction</td>
<td>0.046%</td>
<td>0.004%</td>
</tr>
<tr>
<td>Latency Increase</td>
<td>270 μsec</td>
<td>40 μsec</td>
</tr>
</tbody>
</table>
Conductor Overheads

• Startup overheads
  – ~10 ms per enabled node
  – ~250 μs per null adaptor
• Small for connections that last a few seconds or more
• Offset by the benefits of adaptation
Case Study # 1

Interactive web traffic

Software download

Wireless LAN (2Mb/s)

56Kb/s

100Mb/s

Web Server

Insecure

Low Bandwidth

Encrypt

Prioritize

Encrypt

Introduction — Conductor — Selected Details — Results
Results for Case Study #1

![Bar chart showing download times for Apple, Slashdot, and Photos with and without background traffic.](chart.png)
Case Study #2

Low Bandwidth

56Kb/s (serial)

100Mb/s

56Kb/s (serial)

Compress

Jpeg -> ASCII-art

Introduction — Conductor — Selected Details — Results
Results for Case Study #2

- Download Time (sec)
  - No Conductor
  - Conductor

- Photo Sizes
  - Small (32 * 5 MB)
  - Medium (4 * 215 MB)
  - Large (1 * 895 MB)
Key Properties of Conductor

- Automatic and transparent
  - No user or application action required
- Distributed and coordinated
  - Multiple adaptations at multiple locations
- Incrementally deployable
- Extensible set of adaptations
- Reliable and secure
Other Approaches

- Situation-specific applications
  - Palm clipping apps
  - Text-based web browsers
  » May require specialized applications
  » Requires user diagnosis and intervention
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 [Bellcore/U. Penn]
  – Protocol Transformers
    • Transformer Tunnels [Sudame, Badrinath]
    • Proxy architectures [Fox, Gribble] [Zenel]
  – Active Networks
    » Lack coordination and reliability needed for arbitrary multipoint adaptation
Key Contributions

• Transparent adaptation is desirable and achievable
  – Does not rule out adaptation-aware apps
• Significant benefit to raising the level of services within the network
  – In an incrementally deployable manner
• Reliable delivery of adapted data
  – Allows reliability despite stream modification
Key Contributions

• Security architecture to maintain user control over distributed adaptation
  – With pluggable, self-enforcing authentication
• A working platform for distributed adaptation
  – In daily use
  – A basis for additional research
Conclusions

• Conductor extends adaptation ...
  – Automatic, application unaware
  – Distributed: multi-site, coordinated

• Key enabling services
  – New reliability model: semantic segmentation
  – Framework for automatic planning
  – Security
  – API for adaptation-enabled applications

• Conductor: effective distributed adaptation made easy
Conductor: Distributed Adaptation for Heterogeneous Networks

Mark Yarvis
yarvis@cs.ucla.edu

http://fmg.cs.ucla.edu/Conductor

November 8, 2001