

## Constructing End-to-End Media Paths

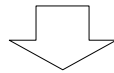
Aki Nakao, Larry Peterson, Andy Bavier

Computer Science Department

Princeton University

## Pervasive Computing Environment

- A wide range of media objects
- An assortment of devices
- A variety of transmission technology



Adaptive, intelligent delivery of media objects

## Motivation

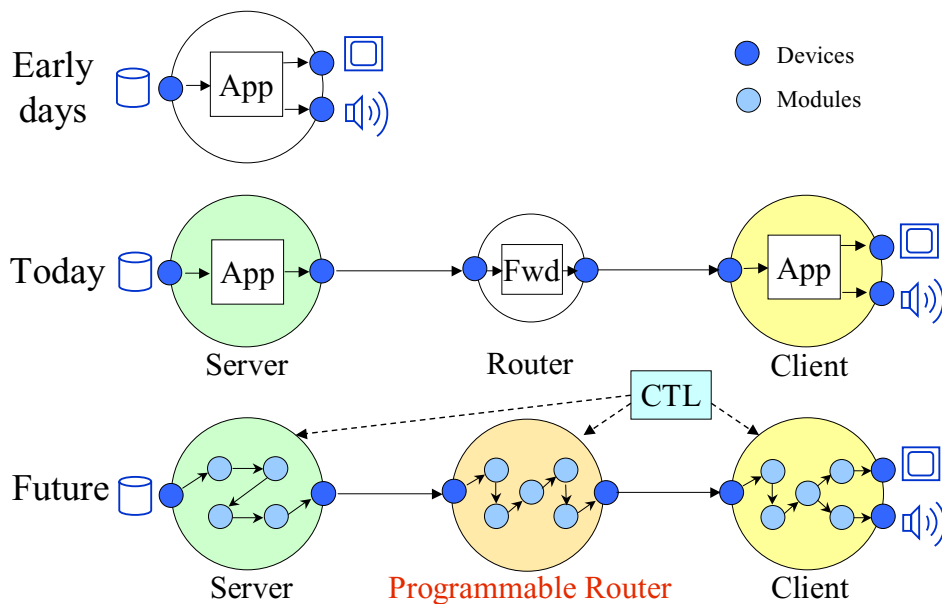
- Introduce degrees of freedom to the current accessing method to media objects

<http://www.video-server.org/movie.mpg>

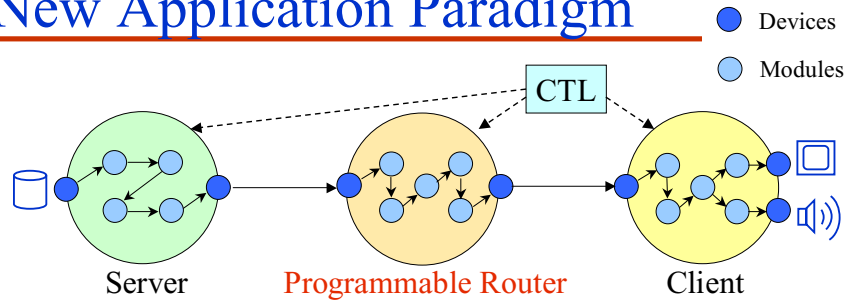
### Limitations

- Services limited to source and sink nodes  
(No intermediate nodes involved)
- Output device on the same node as web-browser
- Client program is monolithic application

## Multimedia Application Evolution



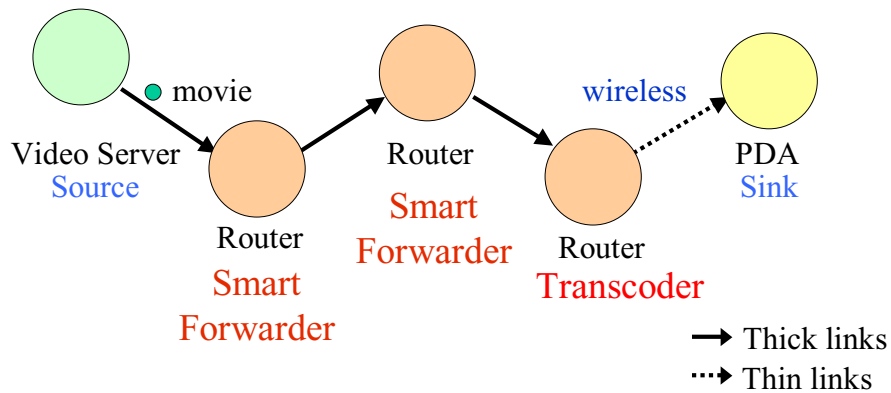
## New Application Paradigm



Degrees of freedom in accessing media objects

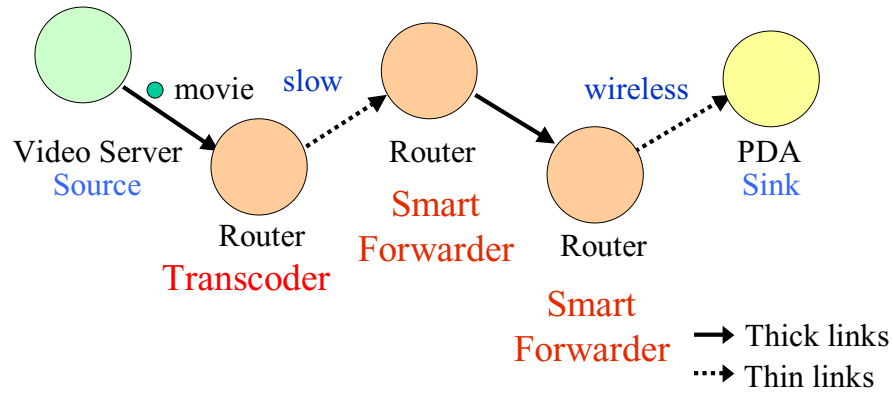
- Adaptation injected in the intermediaries
- Remote construction of distributed applications
- Fine-grained modular applications

## Example Scenario 1



Smart Forwarder : Selectively drop packets in network congestion  
Transcoder : Change stream attributes according to user preference

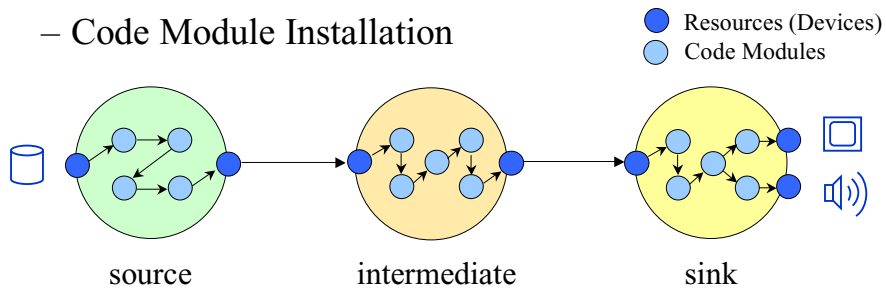
## Example Scenario 2



How to deploy **Function Chains** along the path ?

## End-to-End Media Paths

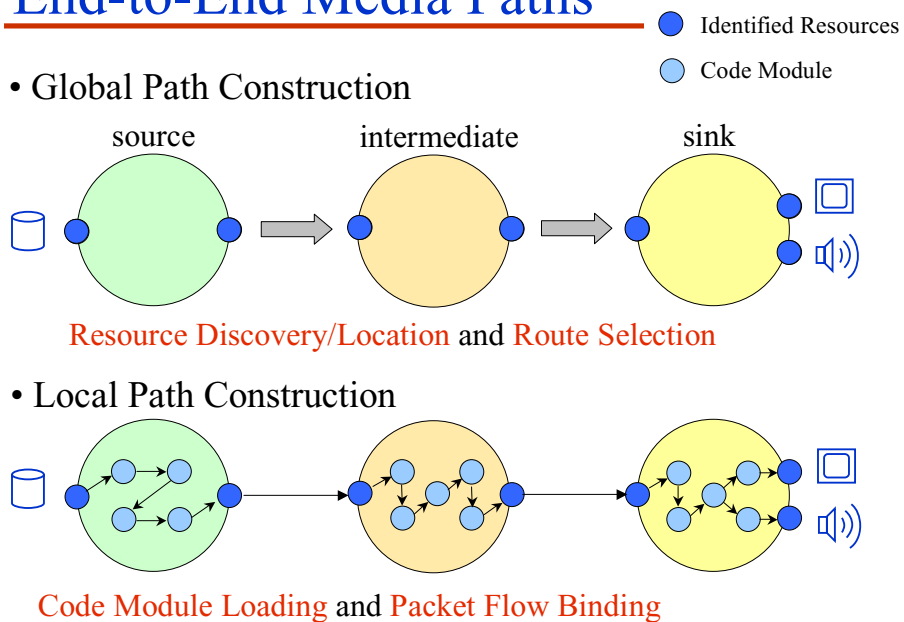
- Framework to string modules from source device to sink device(s)
- Necessary pieces
  - Resource Discovery/Location and Route Selection
  - Code Module Installation



## Information Sources

- Media object
  - Requirements to play
- User
  - Preferences
- Node
  - Capabilities and Devices (Resources)
- Path programmer
  - Rules of Composition (**Path Rule**)

## End-to-End Media Paths

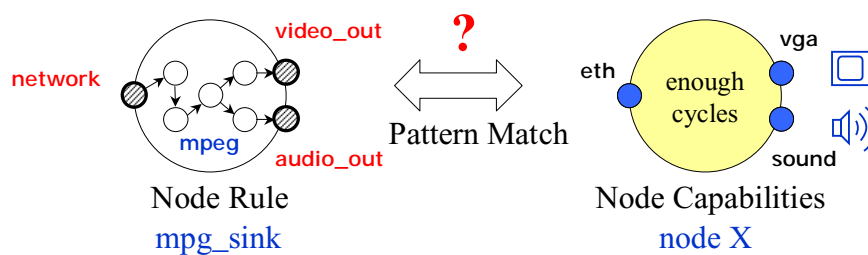


## Global Path Resolution

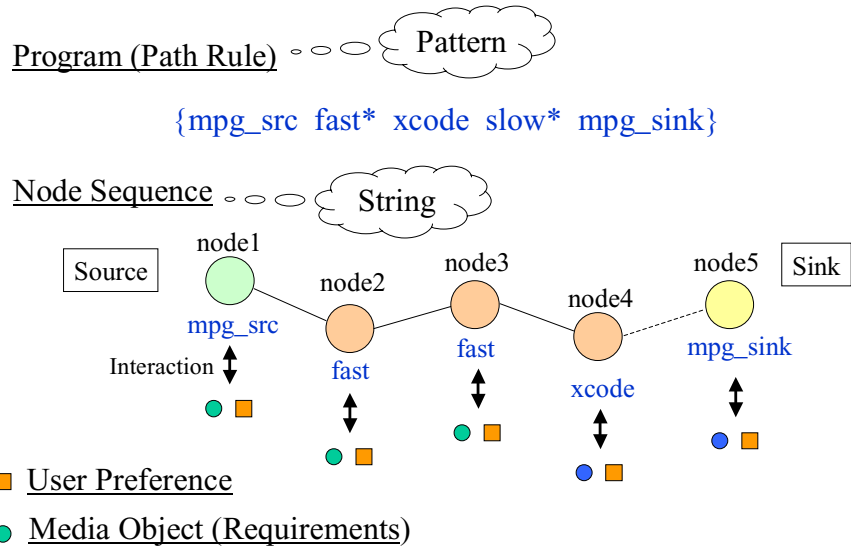
- Resource Discovery
  - Node Capabilities DB
- Resource Location
  - **Pattern Match**  
(**Path Rules** against **Node Capabilities**)
- Routing Selection

## Pattern Matching

- Path Rule (Regular expression of Node Rules)
  - {mpeg\_src fast\* xcode slow\* mpg\_sink}
- Node Rule (Requirements on Node Capabilities)
  - mpg\_src, fast, xcode, slow, mpg\_sink



## Pattern Matching

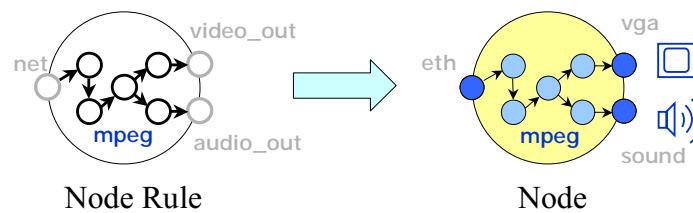


## Global Path Resolution

- Resource Discovery
  - Node Capabilities DB
- Resource Selection
  - Pattern Match
- Routing Selection
  - Shortest Path
  - Exhaustive Search
  - K-shortest paths
  - Overlay

## Local Path Resolution

- Virtual Devices Resolution
- Code Module Download
- Local Path Instantiation



## Prototype

- Global Path Construction
  - Pattern matcher in Java
- Local Path Construction
  - Each node runs Scout OS
  - Extended to support dynamic loading (KMOD)
  - Extended to support path configuration (Ranger)

## Limitations

---

- Security
  - Single Trusted Domain
- Scalability
  - Relatively small scale
- Centralized information sources
  - Node Capabilities and Path Rules stored in Server
  - Single point failure
  - Decision based on stale information
- Response to changing conditions

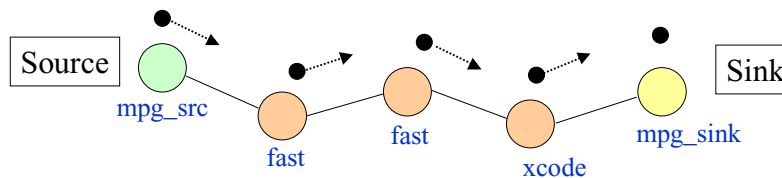
## Distributed Scheme

---

- Lightweight mobile agents (JAVA RMI)
  - DFA of Path Rule visits each node
  - Node capability co-locates with each node

PathRule {mpg\_src fast\* xcode slow\* mpg\_sink}

→ Pattern Matcher (DFA)



## Contributions

---

- Route selection and code selection in one
  - Pattern Match
  - Attribute based (QoS) routing
  - Active Networking
- Isolating the Information Sources
  - Media Object (Requirements)
  - User (Preferences)
  - Node (Capabilities)
  - Path Programmer (Programs)

## End

---

- Publications
  - <http://www.cs.princeton.edu/nsg/publications.html>
- Comments to...
  - [nakao@cs.princeton.edu](mailto:nakao@cs.princeton.edu)