Darwin: Customizable Resource Management for Value-Added Network Services

Prashant Chandra et. al
Carnegie Mellon University

Presented by Jinghua Hu
10/10/2002
Outline

- Introduction
- Integrated customizable resource management
- Darwin Architecture and Mechanisms
- Example
- Summary
Introduction

- Service-oriented network
  - Value-added services in Internet
    - web-caching, video/audio transcoding and mixing, data mining, etc
  - Services operated in a hierarchical fashion
    - applications invoke high-level service providers, which may in turn invoke services from lower level
  - Many challenges are posed
Challenges in Resource Management

- Differences from traditional communication-oriented network
  - A broader set of resources to manage
  - Inter-dependencies between different types of services to be exploited
  - Allow service providers to make resource tradeoffs based on their own notion
  - Resources availability continuously evolves
Service Requirements

- Complex and value-added services
  - Allow service-specific notions of service quality
- Hierarchically structured services
  - Enable reuse of functionality and the need for specialization
Resource Management Requirements

- What resources to manage
  - link capacity, switch capacity, storage capacity, computation power, etc

- Requirements for management
  - **Space**: Resource type and location
  - **Time**: Resource availability over time
  - **Services**: Resource sharing across services
What do we need

- **Customizable** resource management policies along three dimensions
  - space, time, services

- **Integration** of all three dimensions to support value-added services
Darwin Project

- A set of integrated resource management mechanisms that support value-added electronic services

- Four Inter-related Mechanisms
  - High-level Resource Selection: Xena
  - Runtime Resource Management: Delegates
  - Hierarchical Scheduling: Scheduler
  - Low-level Resource Allocation: Beagle
Darwin Architecture

<table>
<thead>
<tr>
<th>Mechanism</th>
<th>Time Scale</th>
<th>Scope of Information and Actions</th>
<th>Customization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Xena</td>
<td>Coarse</td>
<td>Global, High-Level</td>
<td>Domain-Specific Optimizations</td>
</tr>
<tr>
<td>Delegates</td>
<td>Medium</td>
<td>Restricted, Detailed</td>
<td>Customized Policies and Actions</td>
</tr>
<tr>
<td>Hierarchical Scheduler</td>
<td>Fine</td>
<td>Local, Network-Specific</td>
<td>Network Parameters</td>
</tr>
<tr>
<td>Beagle</td>
<td></td>
<td>Coordinates All Other Mechanisms</td>
<td></td>
</tr>
</tbody>
</table>

Figure 2: Complementary nature of the Darwin mechanisms
Resource Representations

- **Virtual Meshes**
  - a core abstraction for resource management
  - a set of resources that are allocated and managed in an integrated fashion to meet the needs of service providers and applications

- **Hierarchical Virtual Meshes**
Virtual Meshes

Figure 1: An Example Resource Management Hierarchy
Darwin Node Software Architecture

Figure 3: Darwin node software architecture
Resource Allocation Procedure

- Three steps
  1. Resource discovery
  2. Solving optimization problems
  3. Allocate resources

- Step 1 and 2 are done by Xena
- Step 3 by Beagle
Part 1: Xena

- High level resource selection
  - resource discovery
  - optimization: max-quality or min-cost

- Input: Application Input Graph
  - nodes: desired services
  - edges: flows connecting specified services

- Output: Virtual Meshes (to Beagle)
Service Specifications

- More constrained specifications
  - network address of service location, services, QoS parameters for flows
  - Xena can only select route flows

- Less constrained specifications
  - leave placement of a service unspecified
  - Xena can place both nodes and route flows
Xena Implementation

- Interfaces
  - Applications
  - Service providers
  - Beagle interface

- Database
  - resource availability database
  - mapping from service/flow types to resources
  - semantics-preserving transformations
Summary: Xena

- Xena provides high-level resource selection
- Performs domain-specific optimization
- Need global information
- Coarse-grained resource management
Part 2: Control Delegates

- Delegates
  - Customizable runtime resource management
  - A code segment sent by applications or service providers to network nodes in order to get customized management for flows
  - An example of active packets!
  - Executed on designated routers
  - Perform operations on subtree flows
Delegates Operations

- **RCI**: Router Control Interface
- Delegate operations supported by RCI
  - Flow definition
  - Resource Management
  - Flow redirection
  - Monitor network status
  - Communication
- Delegates are implemented based on standard JVM
Summary: Delegates

- Delegates provide customizable runtime resource management
- Allow customization on policies and actions in a restricted scope
Part 3: Hierarchical Scheduling

- Implements resource isolation and sharing specified by virtual meshes
- H-FSC Algorithm
  - Hierarchical fair-service curve
  - Allow entities sharing resources at different levels independently manage their own share
Part 4: Beagle

- Beagle: resource allocation protocol
- Input: Virtual Meshes (from Xena)
  - a list of flows and delegates
  - also includes a set of designated routers which identify the mesh core
- Output: flow setup messages
- Upon arrival of Beagle messages, Beagle will
  - pass a resource tree to Local Resource Manager
  - establish delegates on request
Resource Sharing

- Two forms of resource sharing
  - Hierarchical sharing
    - hierarchical group tree
    - flows with the same QoS service type are put in the same group
  - Temporal sharing
    - a set of flows sharing the same resource over time, which may be aggregated
Darwin Demonstration

- Applications:
  - Two video flows, Two simulation flows
- Resources: link bandwidth, computation
- Experiments
  - “abundant bandwidth” scenario
  - “scarce bandwidth” scenario
    - without control delegate
    - with control delegate for video-quality adaptation
Xena

Application flows

(a) Application Input Graph

- Video Source [MJPEG or UNC] → m2
- Video Display [MJPEG or UNC]
- Video Source [UNC only] → m6
- Video Display [MJPEG or UNC]
- Simulation [app-specific type]
- Simulation [app-specific type]
- Simulation [app-specific type]
Beagle

Virtual Application Mesh

(b)

<table>
<thead>
<tr>
<th>Flow</th>
<th>QoS Type</th>
<th>FlowSpec</th>
</tr>
</thead>
<tbody>
<tr>
<td>F₁</td>
<td>Guaranteed</td>
<td>((p, r, b) = (1, 1, 0)) (R, S) = (1, 0)</td>
</tr>
<tr>
<td>F₂</td>
<td>Guaranteed</td>
<td>((p, r, b) = (1, 1, 0)) (R, S) = (1, 0)</td>
</tr>
<tr>
<td>F₃</td>
<td>WFS</td>
<td>(W = 10)</td>
</tr>
<tr>
<td>F₄</td>
<td>WFS</td>
<td>(W = 10)</td>
</tr>
</tbody>
</table>

Temporal Sharing

<table>
<thead>
<tr>
<th>Flow Group</th>
<th>Aggregate QoS Type</th>
<th>Aggregate FlowSpec</th>
</tr>
</thead>
<tbody>
<tr>
<td>F₃ and F₄</td>
<td>Controlled Load</td>
<td>((p, r, b) = (10, 5, 128))</td>
</tr>
</tbody>
</table>
Local Resource Manager

(c) 

\[ L_2: \text{Whiteface} \rightarrow R1 \]

\[ L_1: \text{Timberline} \rightarrow \text{Whiteface} \]
Bandwidth Sharing

- Without delegate (b), video flows are sent at the lowest quality level, even when simulation is idle.
- With delegate (c), video flows get better sharing when simulation is idle.

Figure 10: Bandwidth sharing within the example application: a) abundant bandwidth scenario; b) scarce bandwidth scenario without the control delegate; c) scarce bandwidth scenario with the control delegate.
Put it all together
Summary

- Contributions
  - Design of Darwin, a customizable integrated resource management system for service-oriented network
  - Four inter-related resource management mechanisms work together or plug-and-play
  - Scalability and Security issues remain