A Model-driven, Component-based and Service-oriented Approach for Designing an Autonomic Transport Protocol

Ernesto Expósito
ernesto.exposito@laas.fr

Université de Toulouse
LAAS-CNRS
INSA
Context and problem statement

Distributed applications requirements

Transport protocols

Network services and technologies

New generation transport layer

Which service?
- Application reqs + network const.
  - monolithic vs composite?

How to adapt?
- Component behavior
  - Composite struct.

How to be autonomous?
- Self-config
  - Self-adapt

- TCP
- DCCP
- SCTP
- MPTCP
- Bandwidth
- Time constraints (delay, jitter, synchronization)
- Partial reliability
- Partial order

- Best-Effort
- IntServ
- DiffServ
- MPLS
- Wired
- High speed
- Wireless/Mobility
- WiFi
- 3G

reliability order
# Approach

## Methodology

requirements-oriented, model-driven, unified process:

1) Requirements
2) Design
3) Specification (model) and validation
4) Implementation
5) Tests and performance evaluation
6) Deployment

## Models

- **Software architecture:** UML
- **Knowledge:** OWL (ontologies)
- **Decision:** analytic/learning-based model

## Paradigms

- **Component-based design** (vs. monolithic)
- **Adaptive and autonomic management** (vs. manual-human mg.)
- **Service-oriented design** (vs. static/hard-binding)
Outline

- New generation transport layer (incremental design)
  - Application and network aware
  - Component-based and (micro)service-oriented
  - Adaptive
  - Autonomic
Phase 1: the basis

- Requirements
- Design of the Fully Programmable Transport Protocol (FPTP)
- Specification and validation
- Implementation and test/performance evaluation
- Deployment
Requirements

- R1: How to provide the most adequate service taking into account application requirements and network services
- R2: How to easily integrate future components (more specialized mechanisms)
Design (RI): mechanisms

- Reliability
  - Full/Partial reliability (FR/PR)
  - Rate Control (RC)
  - Network Resources (EF/AF)
  - Network Congestion (BE)

- Differentiation (i.e. I,P,B images)
  - Differentiated Partial reliability (D-PR)
  - TCP-Friendly Rate Control (TFRC)

- Differentiation and Time-constrained Partial reliability (TD-PR)
  - TCP-friendly rate control (TD-TFRC)

Composition of mechanisms (reliability, order, time, network resources):

- {PR,D-PR,TD-PR} x {RC,TD-RC,TFRC, TD-TFRC}

- Time constraints (delay, jitter, synchronization, bandwidth)

- Application aware
- Application & network aware
- Network aware
Design (R2): architecture
Specification: transport mechanism composition

Partially reliable function (PR)
Specification: mechanism active behavior
Validation of UML specification

- Environment: IBM-Rational TAU platform (profile: UML-verification)
  - Generation of executable model

- Approach
  - Dynamic model consistency
  - Interactive simulations for functional validation:
    - Validation per use case: instantiation, interconnection, communication, deadlocks free
  - Limitations: complexity to cover all potential protocol states
Implementation and Test/performance evaluation

- JAVA implementation
- Evaluation:
  - Experimental network environment based on a network emulator (Dummynet) and streaming audio/video applications
Deployment

• European Project GCAP (1999-2001):
  • Active deployment of FPTP services (active networks)

  • Deployment and evaluation of application-aware/network-aware mechanisms over heterogeneous network services
Incremental design....

- Benefits of FPTP by offering
  - A component-based architecture
  - A large set of application and network-aware composite transport mechanisms
- New requirements: more elaborated adaptive strategies
Phase II: adaptation

- Requirements
- Design of the Enhanced Transport Protocol
  - Adaptive: Behavioral and structural adaptation
- Specification and validation
- Implementation and test/performance evaluation
- Deployment
Requirements

• Req 3: Behavioral adaptation
  • Adaptive mechanisms based on generic application traffic semantic

• Req 4: Structural adaptation
  • Dynamic configuration of compositions in response to network changes
    • learning based decision model
Design of a model-driven QoS interpreter to allow generic behavioral adaptation (R3)

- Generic interpreter (vs. ad-hoc solutions) offering standard interface to retrieve properties/constraints of multimedia streams (i.e. H.264, MPEG2, H.263, etc.)
- Used for designing/developing QoS adaptive mechanisms (i.e. error control, rate/congestion control)
Design of model-driven structural adaptation (R4)

- Analytic model approach
  - Includes all the valid compositions
  - Guides the selection based on requirements and network conditions
- Learning-based model approach
  - Extension of the Markov Decision process (eMDP)
  - Obtained by reinforcement learning techniques
Interactive video-conferencing (I,P picture) application (max 150 ms delay)

Behavioral adaptation (retransmission-based mech)

(I,P)=(100%,50%)
(I,P)=(100%,0%)
(I,P)=(50%,0%)

Structural adapt.

FEC Redundancy (I,P)=(150%,0%)

Accumulated delay (ms)

% PLR (emulated network, e2e delay=40ms)

(I,P)=(50%,0%)
(I,P)=(100%,0%)
(I,P)=(100%,50%)
Behavioral and Structural adaptation (TFRC)

Traffic profile for a H.263 video stream

Comm. channel simulating congested, delayed and lossy network scenarios

Video data received by

QoS improv. 15%-56% (PSNR)
Deployment

  - Behavioral adaptation strategies (RC) over heterogeneous network services
  - Integration of adaptive ETP services within the autonomic NetQoS system
Incremental design....

- Adaptive transport protocol mature for autonomic
  - Behavioral adaptation
  - Structural adaptation
- New requirements:
  - Self-configuring
  - Self-adapting
Phase III: autonomic

- Requirements
- Design of the Autonomic Transport Protocol
  - Autonomic, service-oriented and ontology-driven architecture
- Specification and validation
- Implementation and test/performance evaluation
- Deployment
- Conclusions
Requirements

• Req 5: Integration of current solution within an autonomic architecture

• Req 6: AC knowledge base for self-configuring and self-adapting properties
Specification of the Autonomic Transport Protocol Architecture (R5)
Implementation of an Ontology-driven Autonomic-manager offering self-configuring and self-adapting functions (R6)

**ODA Ontologies**

**V1**: representation and consistency: transport mechanisms, functions, protocols and services

**V2**: inferencing capabilities for SOA/CB self-configuring: services properties, components and composites

**V3**: inferencing capabilities for self-adapting behavioral (tuning) structural (reconfiguring)

**QoS Transport Ontology**

OWL implementation
Example: Error/throughput/time controlled service discovery

<table>
<thead>
<tr>
<th>OWL-Class:</th>
<th>Error_throughput_and_time_controlled_service</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intersection of:</td>
<td>Time_controlled_service</td>
</tr>
<tr>
<td></td>
<td>Error_controlled_service</td>
</tr>
<tr>
<td></td>
<td>Throughput_controlled_service</td>
</tr>
<tr>
<td>Subclass of:</td>
<td>Time_controlled_service (Why?)</td>
</tr>
<tr>
<td></td>
<td>Error_and_throughput_controlled_service (Why?)</td>
</tr>
<tr>
<td>Instances:</td>
<td>FPTP (Why?)</td>
</tr>
<tr>
<td></td>
<td>ETP (Why?)</td>
</tr>
</tbody>
</table>

**Axioms causing the inference**

1. `(Error_throughput_and_time_controlled_service = (Time_controlled_service ∩ Error_controlled_service ∩ Throughput_controlled_service))`
2. `(_(Time_controlled_service = (Transport_service ∩ (Implements . Time_control_function))))`
3. `(_(FPTP rdf:type Transport_service))`
4. `(_(FPTP implements FPTP_TD_RC))`
5. `(_(FPTP implements FPTP_FR))`
6. `(_(FPTP_FR rdf:type Fully_ordered_control_function))`
7. `(_(Fully_ordered_control_function ⊆ Error_control_function))`
8. `(_(FPTP_TD_RC rdf:type Rate_control_function))`
9. `(_(Rate_control_function ⊆ Throughput_control_function))`
10. `(_(FPTP_TD_RC rdf:type Time_control_function))`
11. `(_(Error_controlled_service = (Transport_service ∩ (Implements . Error_control_function))))`
12. `(_(Throughput_controlled_service = (Transport_service ∩ (Implements . Throughput_control_function))))`
Deployment

• European (Celtic) Project Feel@home (2008-2010):
  • Design and development of an Ontology-driven architecture for autonomic QoS management in home networks (UPnP)

• IMAGINE (starting from 2011) IP Project (Virtual Factories/Enterprises)
  • Design and development of an autonomic service bus integrating ATP services for heterogeneous information systems
Conclusions and Perspectives
Conclusions and Perspectives

App/Net aware & Component based design

Behavioral and structural adaptation

V1: FPTP

V2: ETP

V3: ATP

QoS Transport Ontology

Transport layer semantic model (ODA)

Autonomic Computing

ATP orchestration

ODA self-*