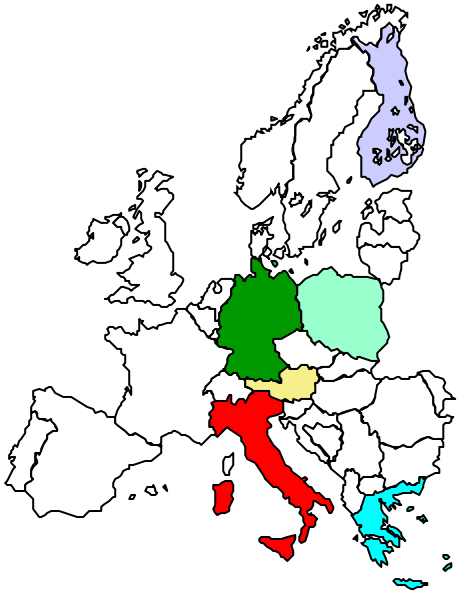
AQUILA (IST-1999-10077)

Adaptive Resource Control for QoS Using an IP-based Layered Architecture



Project Review No. 4
Premium IP Cluster Workshop

Maastricht, The Netherlands
May 15, 2002

http://www.ist-aquila.org/

Outline

- Project Overview
  - *Bert F. Koch (Siemens)*
- Dynamic IP QoS:
  To be or not to be (scalable)?
  - *Stefano Salsano (CoRiTeL)*
- Application Profiles
  - *Anne Thomas (TU Dresden)*
- Quality of Service
  Management Tool
  - *Yannis Karadimas (Q-Systems)*

Consortium

Siemens, Germany

Bertelsmann, Germany

Dresden Univ. of Technology, Germany

Salzburg Research,

T-Systems Nova, Germany

Polish Telecom, Poland

NTUA, Greece

Elisa Communications, Finland

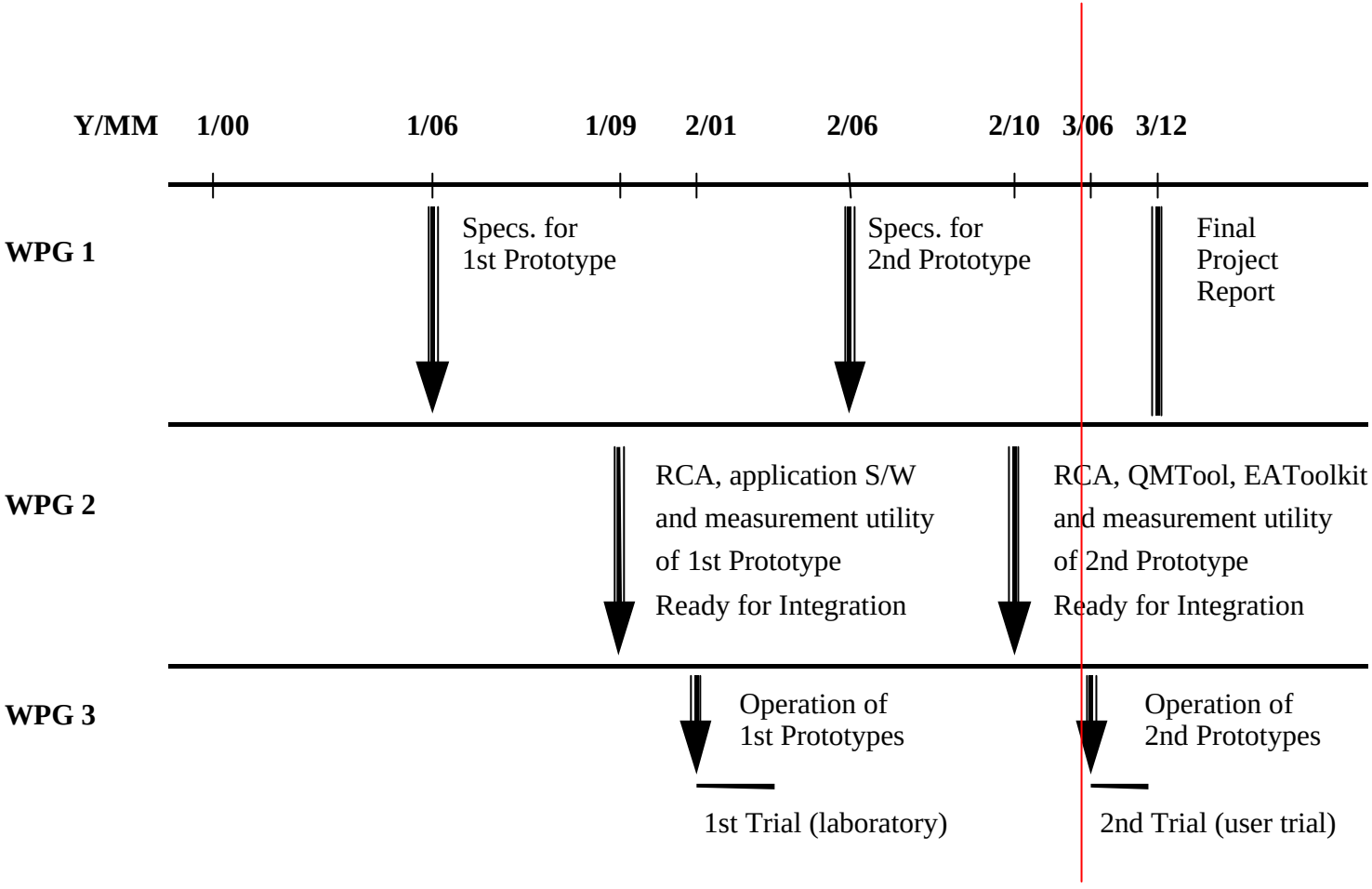
CoRiTeL, Italy

Q-Systems, Greece

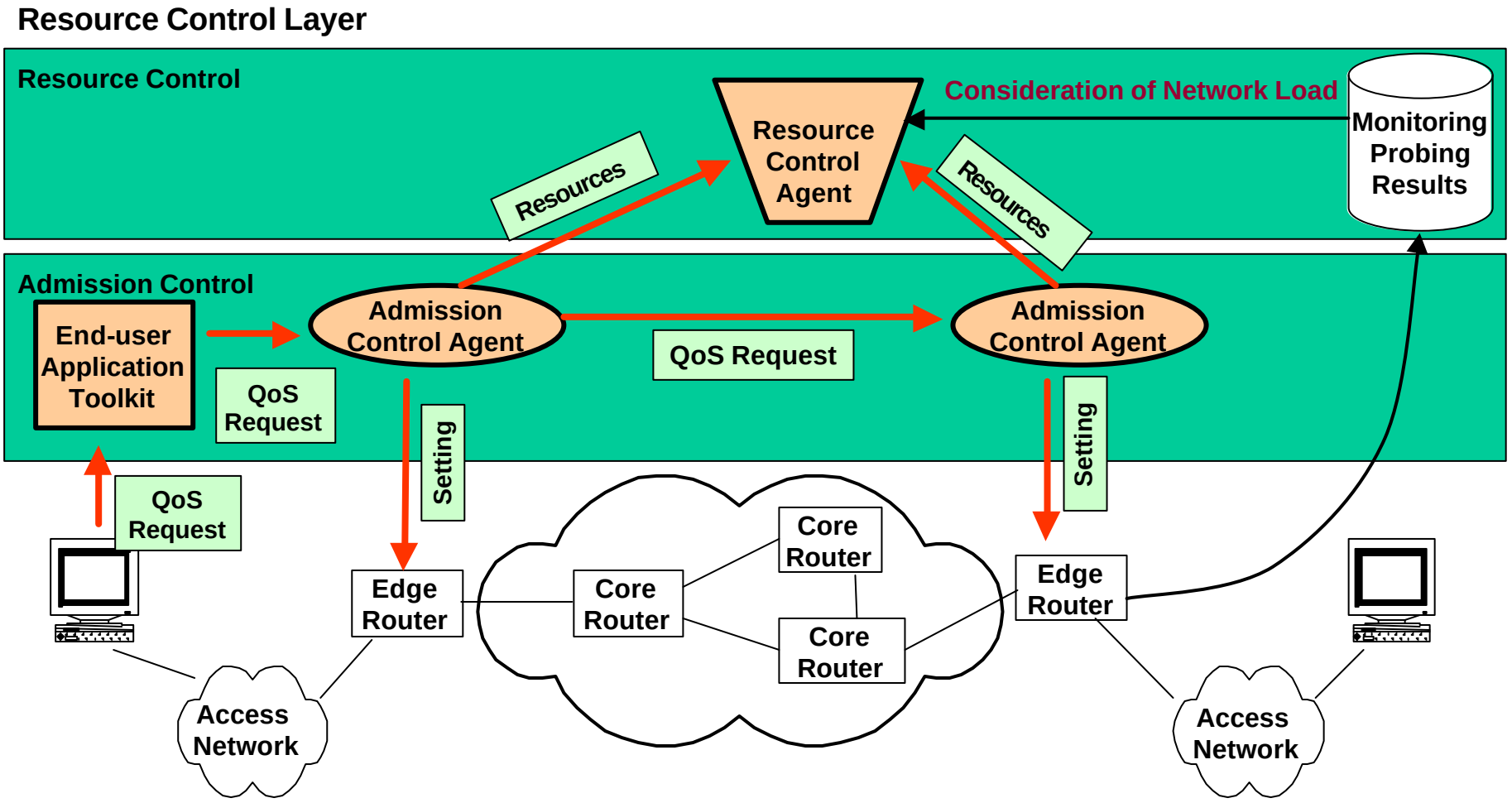
Telekom Austria, Austria

Warsaw Univ. of Technology, Poland

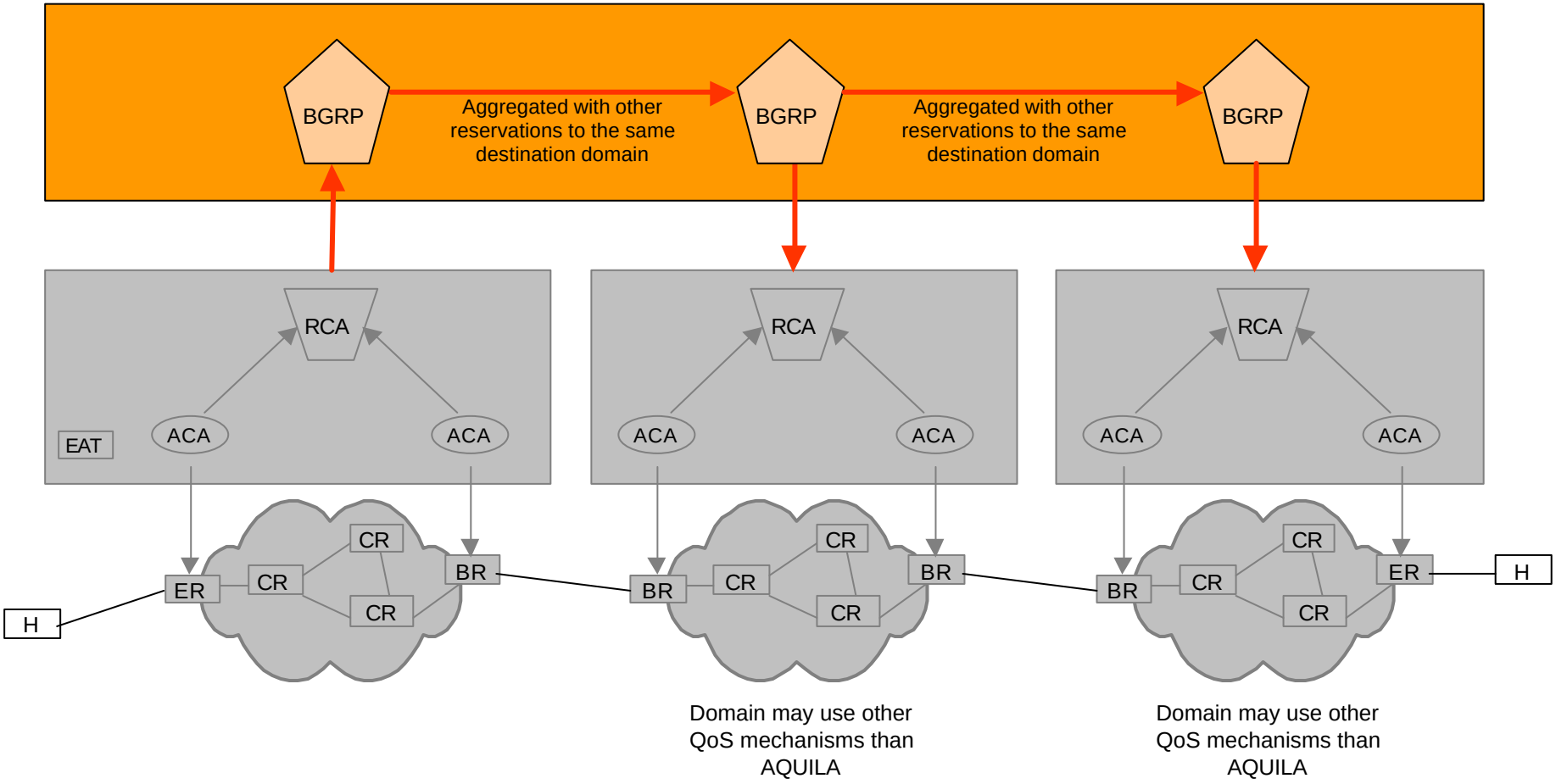
Project Schedule



Architecture



Inter-domain QoS



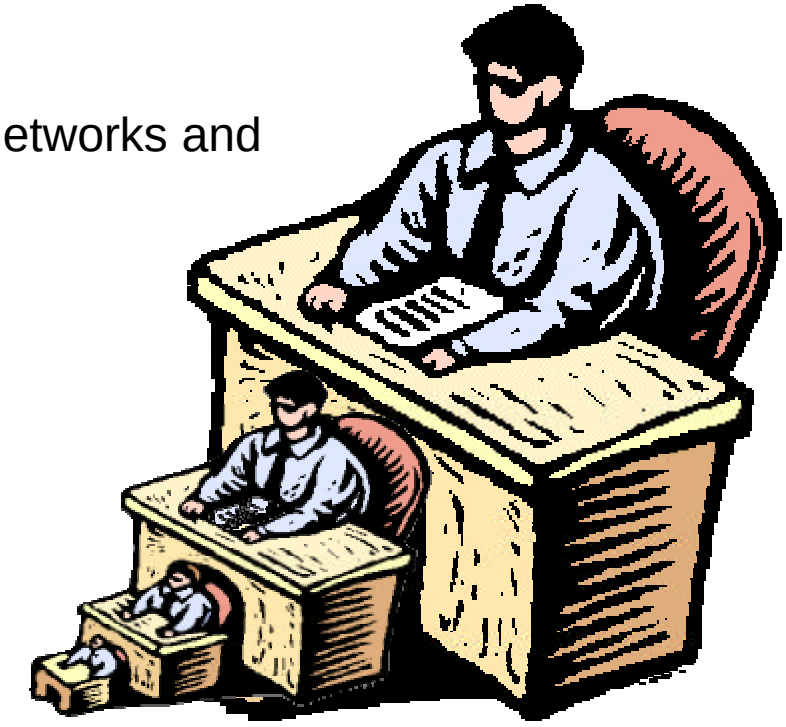
Dynamic IP QoS:
To be or not to be (scalable)?

Outline

- **Scalability & IP QoS scenario**
- Intra-domain scalability - Modelling the problem
- AQUILA solutions: distributed ACAs and Resource Pools
- Inter-domain QoS and scalability
- Conclusion

What is Scalability for IP QoS?

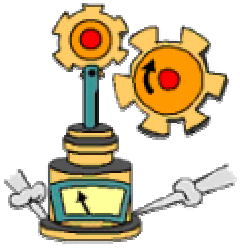
- **The possibility to provide IP QoS services in real-life networks**
  - Taking into account the size of real networks and management/operational issues



- **In particular we are covering <u>dynamic</u> IP QoS**

Scalability of Dynamic IP QoS

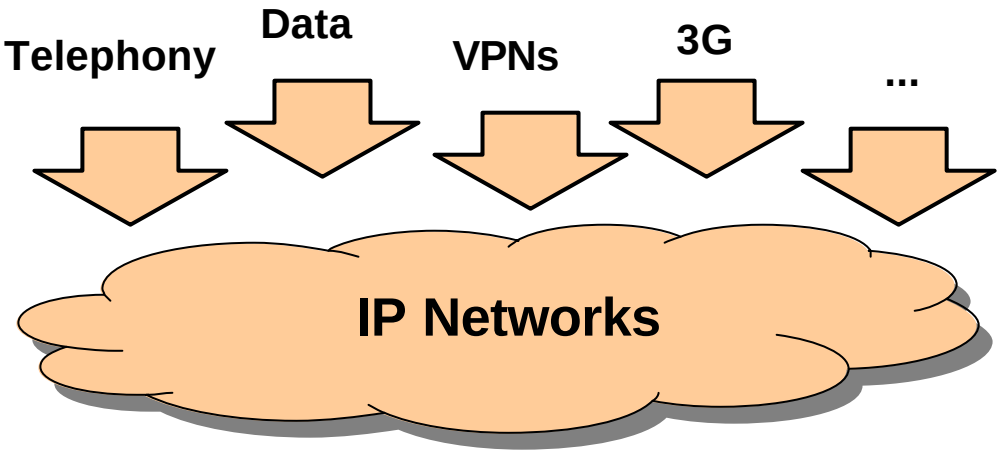
- **Technical feasibility vs. economical analysis**



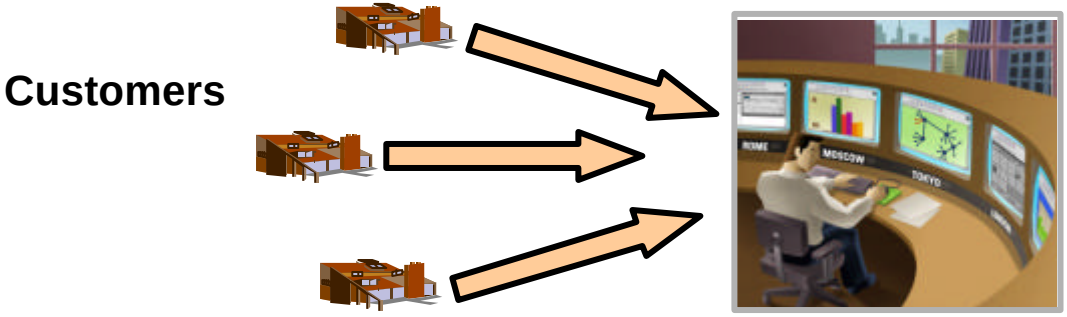
- AQUILA is mainly covering the first issue

General Trends

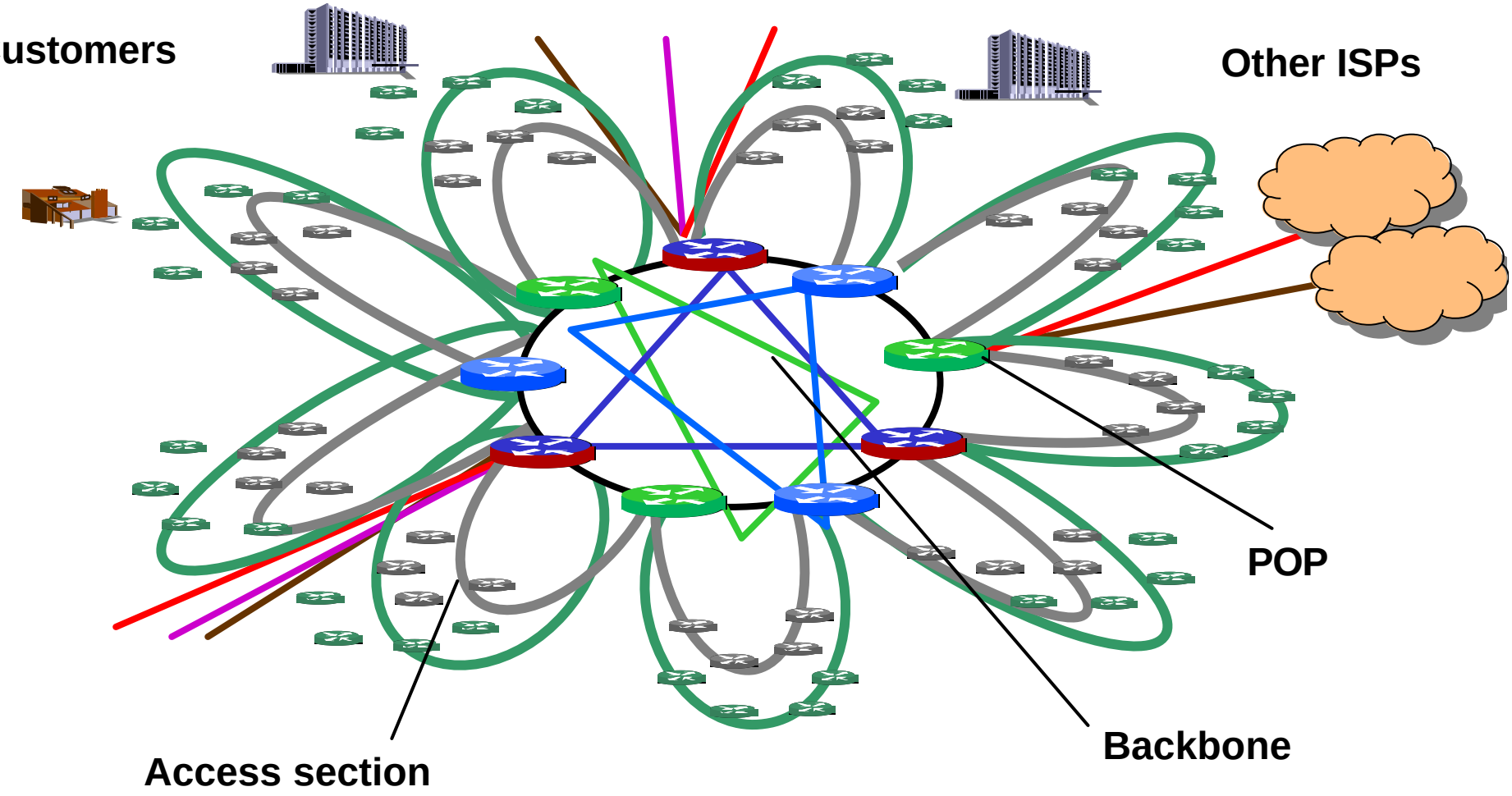
- **Platform convergence**
  - Everything on IP



- **Outsourcing of functionality towards the network operator**
  - From Leased lines to ATM to IP/MPLS



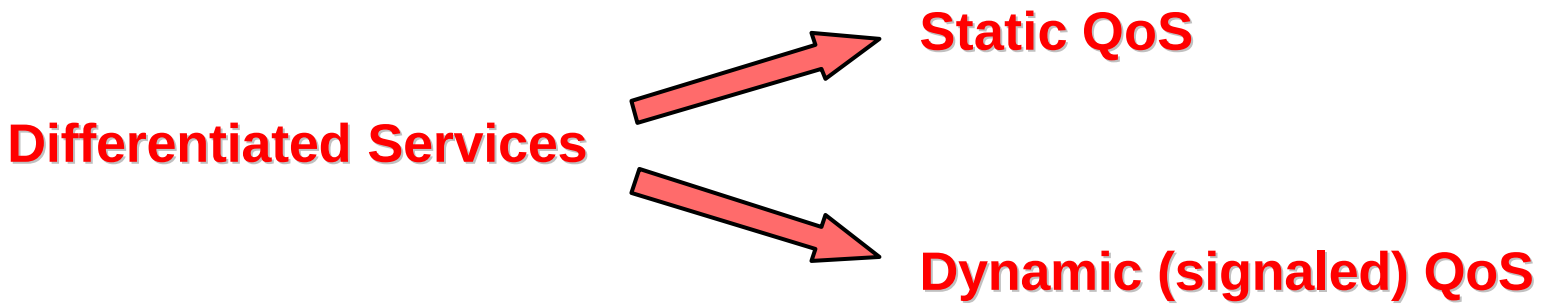
Typical Architecture of a Big Provider



Solutions for QoS

- **Over-provisioning / Traffic engineering**
  - Ensuring service quality… but no IP QoS in strict sense

- **Strict sense IP QoS:**



Do we need more Sophisticated Solutions than Over-provisioning? Yes!

- **On network bottlenecks**
  - Greedy TCP applications always create bottlenecks … (“strict sense” IP QoS may be indispensable for end-to-end QoS services)
  - Access sections:
    - provider access links
    - Customer Edge to Provider Edge
    - customer links
- **In case of failures**
  - It could be unreasonable to over-engineer the networks, also taking into account failure situations
- **To prevent from the case of mis-provisioning**

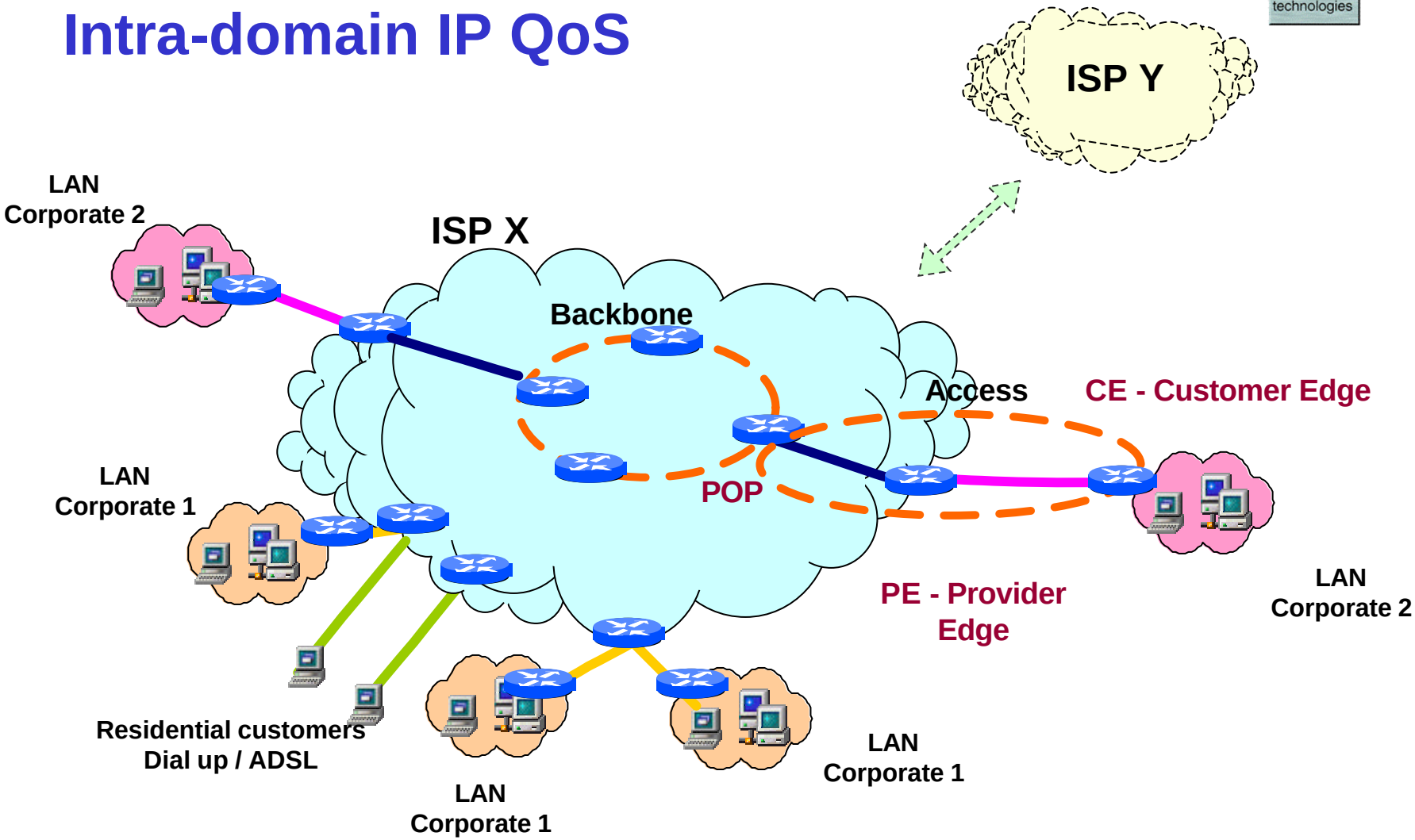
Static vs. Dynamic IP QoS

- **If we accept the idea of having QoS in the network, we should choose whether to have it dynamic or not**
- **In general: the more dynamic … the more efficient, but we should consider the scalability aspects**
  - Can we stand the load of reservation requests ?
  - Can we handle the reservation state in nodes ?
  - Can we manage this more complex network ?
  - What is the complexity of billing ?
  - What is the business model ?

Outline

- What is scalability & IP QoS scenario
- **Intra-domain scalability - Modelling the problem**
- AQUILA solutions: distributed ACAs and Resource Pools
- Inter-domain QoS and scalability
- Conclusion

Intra-domain IP QoS



How to Model the Problem

- **Backbone**
  - 20 POP, each pop several 600 Mbit/s or 2 Gbit/s links and several access routers
- **ISP Access Routers (AR)**
  - Several business customer links, up to hundreds or thousands of residential customers
- **Business customer links**
  - 2 Mbit/s - 34 Mbit/s
- **Residential access**
  - Modem (50 Kbit/s), ADSL (500 Kbit/s), ethernet to the home (10 Mbit/s)
- **What is a flow:**
  - Phone call 8 Kbit/s, video 500 Kbit/s
  - ...

"Scalability Model": Number of active flows and Signaling load (as Reservations/sec)

Backbone lines / BIG POP	10
Backbone line capacity	2,5 Gbit/s
Access routers / BIG POP	10
Access router agggregate cap.	600 Mbit/s

Flow size	Flows/ Big POP	Flows/ AR	Flow duration	Res./ Big POP/ sec	Res./ AR/ sec
8 Kbit/s	1625000	40625	**200 s**	8125	203
500 Kbit/s	26000	650	**1000 s**	26	0.650
2 Mbit/s	6500	162.5	**5000 s**	1.3	0.033

AQUILA Solutions … coming next

- **Distribute the signaling load and the state information towards the edge**
  - Local ACAs / centralized RCA
  - Resource pools
- **This means more scalability and could imply weaker quality guarantees and/or less efficiency**
- **We have to make sure that the quality/efficiency trade-off is satisfactory**

Outline

- What is scalability & IP QoS scenario
- Intra-domain scalability - Modelling the problem
- **AQUILA solutions: distributed ACAs and Resource Pools**
- Inter-domain QoS and scalability
- Conclusion

Distributed AC and Resource Pools

- The Admission Control is distributed, using a "Provisioning" model with re-arrangement
- "Resource pools" are used to dynamically control resource usage

A Two Layer Resource Management

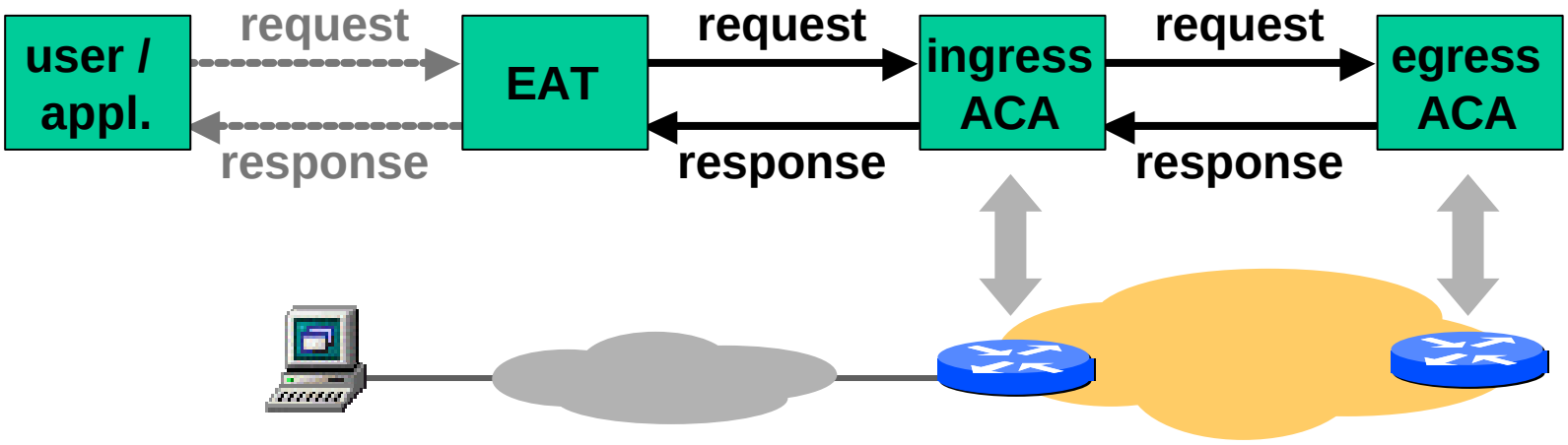
- **AQUILA uses a two layer approach**
  - Combines scalability and efficiency in terms of bandwidth utilisation
- **1st layer: Distributed AC**
  - Distributed and independent local AC decisions at the network edge based on provisioned resource shares (bandwidth)
  - Scalability
- **2nd layer: Dynamic Resource Provisioning**
  - Dynamic re-arrangement of resource shares provisioned to ERs based on demand measurements through a resource pool
  - Resource pool hierarchies
  - Efficiency

Distributed Intra-domain AC

- **AC at the network edge**
  - Admission control agent (ACA) per edge router (ER)
    - per flow AC (micro-flows as well as aggregated flows)
    - ingress AC, egress AC
    - provisioned bandwidth describes available network resources for AC
    - dynamic re-arrangement of provisioned bandwidth through resource pools
  - ACA per border router (BR)
    - same intra-domain AC as on each ER
- **No AC at core routers**
- **No centralised Bandwidth Broker**

Signalling Messages of Intra-domain AC

- **Two request messages per flow**
  - Two independent processing steps at the network border
    - ingress ER
    - egress ER
  - Signalling load scales with size of ER not with network size

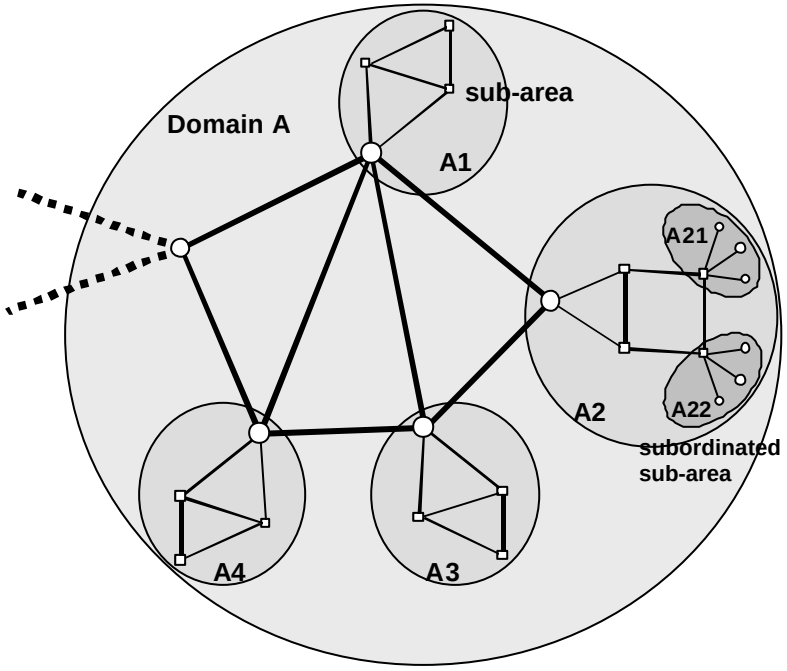


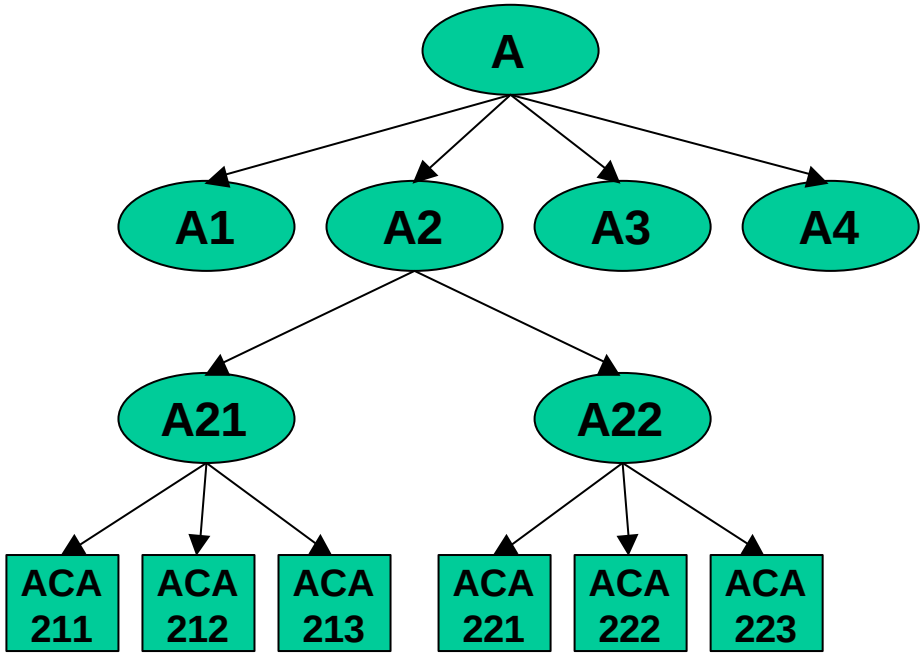
State Variables of Intra-domain AC

- **Independent state variables per ER**
- **Provisioned bandwidth per Traffic Class (TCL)**
- **Bandwidth in use per TCL**
- **DBAC (declaration based admission control)**
  - Uses per flow traffic descriptor (peak rate, sustainable rate)
- **MBAC (measurement based admission control)**
  - TCL1: some aggregated peak rates only (→ ageing window)
  - TCL2: peak rate per flow

Resource Pools (1)

- Resource pools (RP) are used to share common network resources between ACAs
- Hierarchical RP trees





Resource Pools (2)

- **ACAs request more bandwidth from their RP if needed and return unused bandwidth to their RP**
  - Long term demand estimation
  - Based on demand measurement
  - Trade-off between re-assignment frequency and utilisation of assigned bandwidth

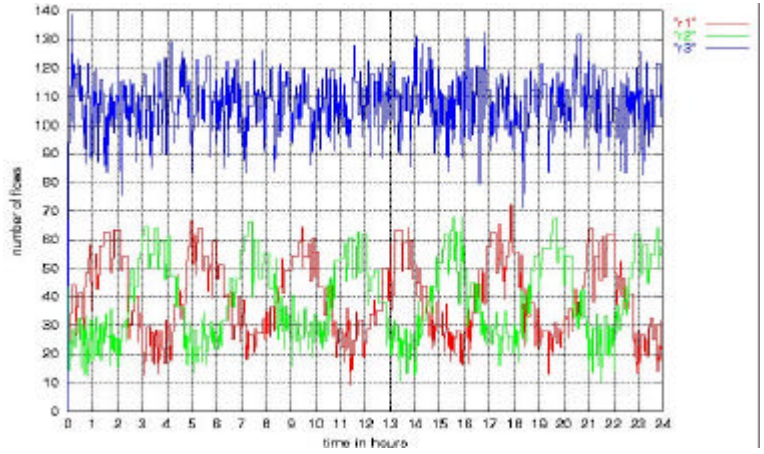
- **Scalability in terms of signalling load**
  - Bandwidth is re-assigned not per flow but based on long-term demand estimation
  - Different time scale than AC (less frequently)
  - We achieved two orders of magnitude in time scale (simulation, VoIP traffic model)

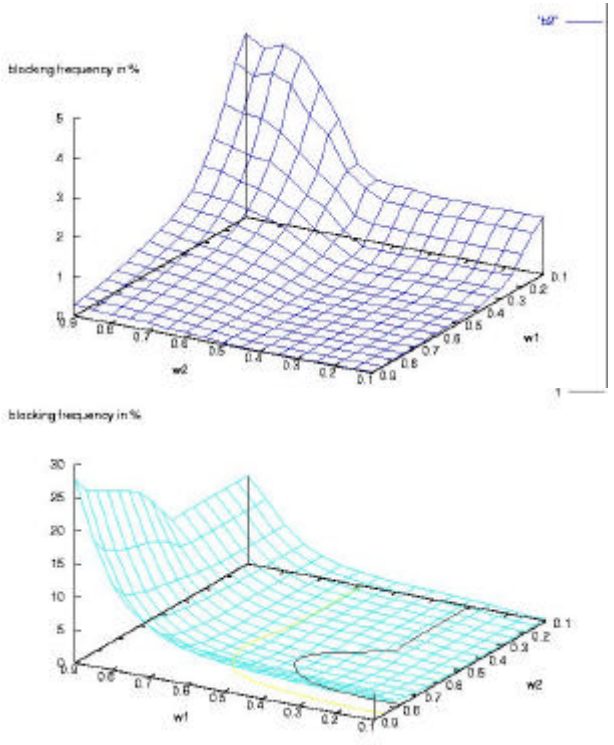
Resource Pools (3)

- **Scalability in terms of state variables**
  - State variables per RP
    - available bandwidth per TCL
  - State variables per ACA
    - available bandwidth per TCL
    - some few parameters used for demand estimation

Distributed AC and Resource Pools: Performance Evaluation

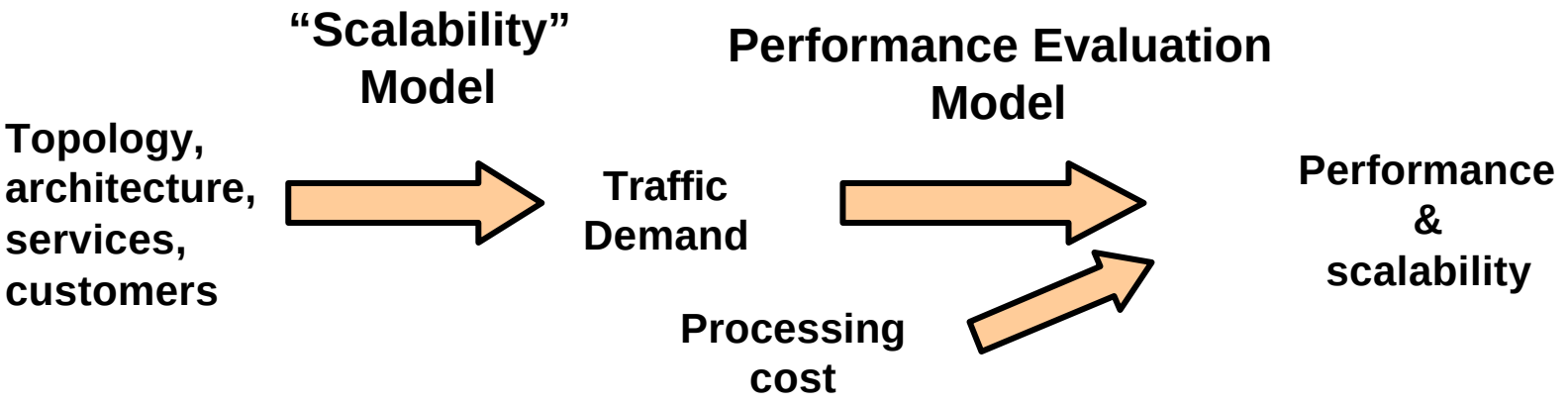
- **Given a traffic demand model:**
  - Distribution of reservation requests inter-arrival time
  - Distribution of flow size
- **We evaluate performances**
  - Reduction of signalling load
  - Efficiency in resource utilisation





Evaluate the Scalability

- Using the “scalability model” we try to identify the number of transaction/second for Resource Control Layer elements under various combinations of offered QoS traffic
- From our prototype implementation we try to gather experience on the processing cost of various transactions (of course this is biased by our design/implementation choices … but it is an indication)



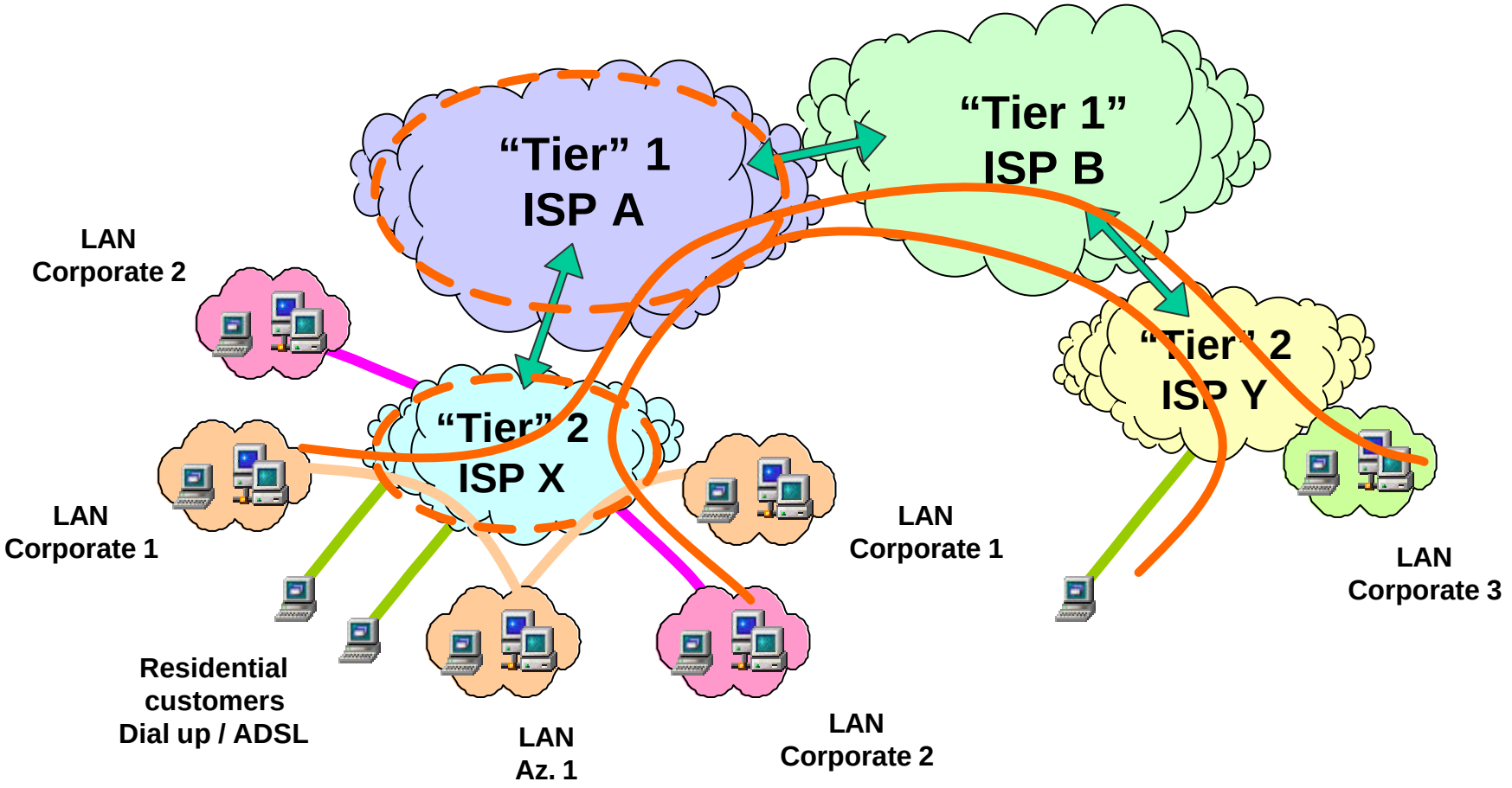
"Operational Scalability"

- This refers to the additional procedures needed to configure/manage the QoS networks
- In the 1st trial we had to manually configure the mapping of destination IP address into egress Edge Router - Now we automatically use BGP next hop information
- We currently need to manually configure the weights for WFQ
- In order to communicate with the routers we are using "telnet" and CLI (Command Line Interface) … this has some scalability limitation

Outline

- What is scalability & IP QoS scenario
- Intra-domain scalability - Modelling the problem
- AQUILA solutions: distributed ACAs and Resource Pools
- **Inter-domain QoS and scalability**
- Conclusion

Inter-domain IP QoS



Inter-domain Issues

- Inter-domain dynamic IP QoS is even more challenging
- As with intra-domain, economical aspects / business models need to be analysed ...
- From the technical viewpoint … we defined the BGRP+ solution

BGRP Plus (BGRP+)

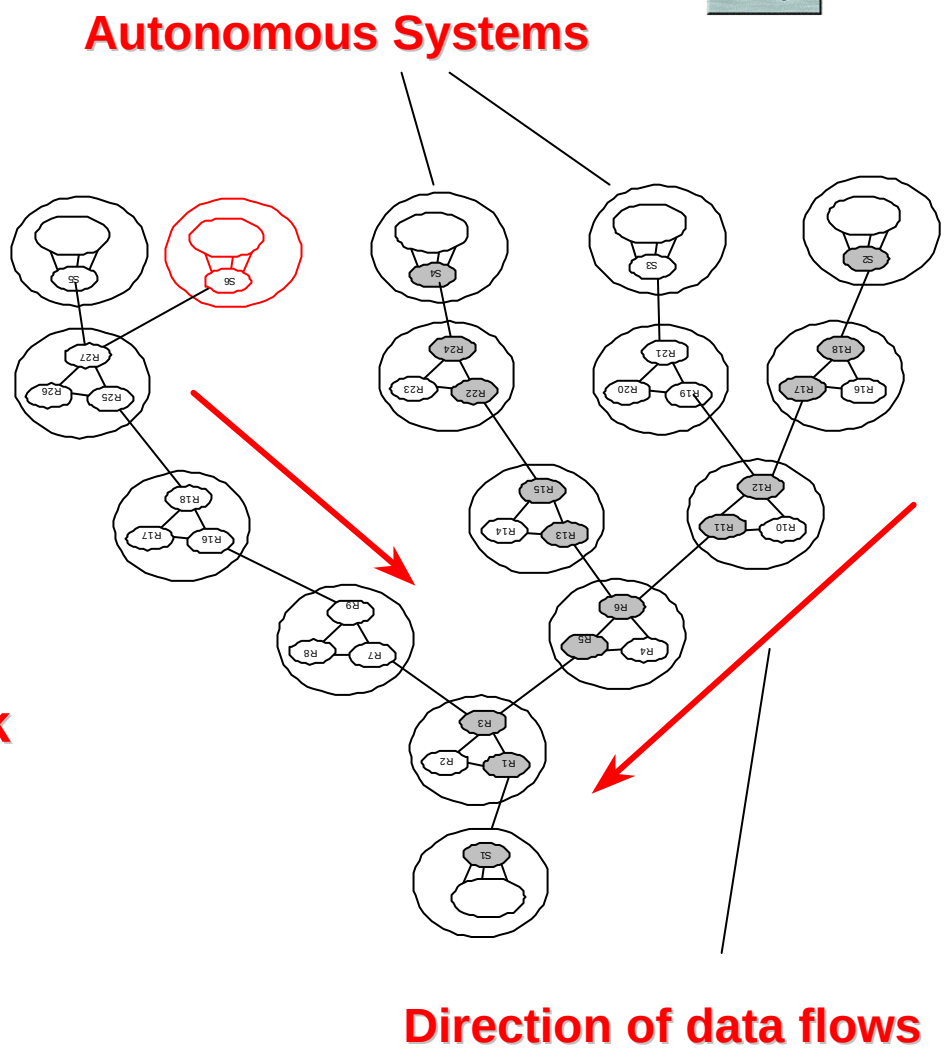
- BGRP+ addresses scalability in terms of

  State information ("Sink tree")

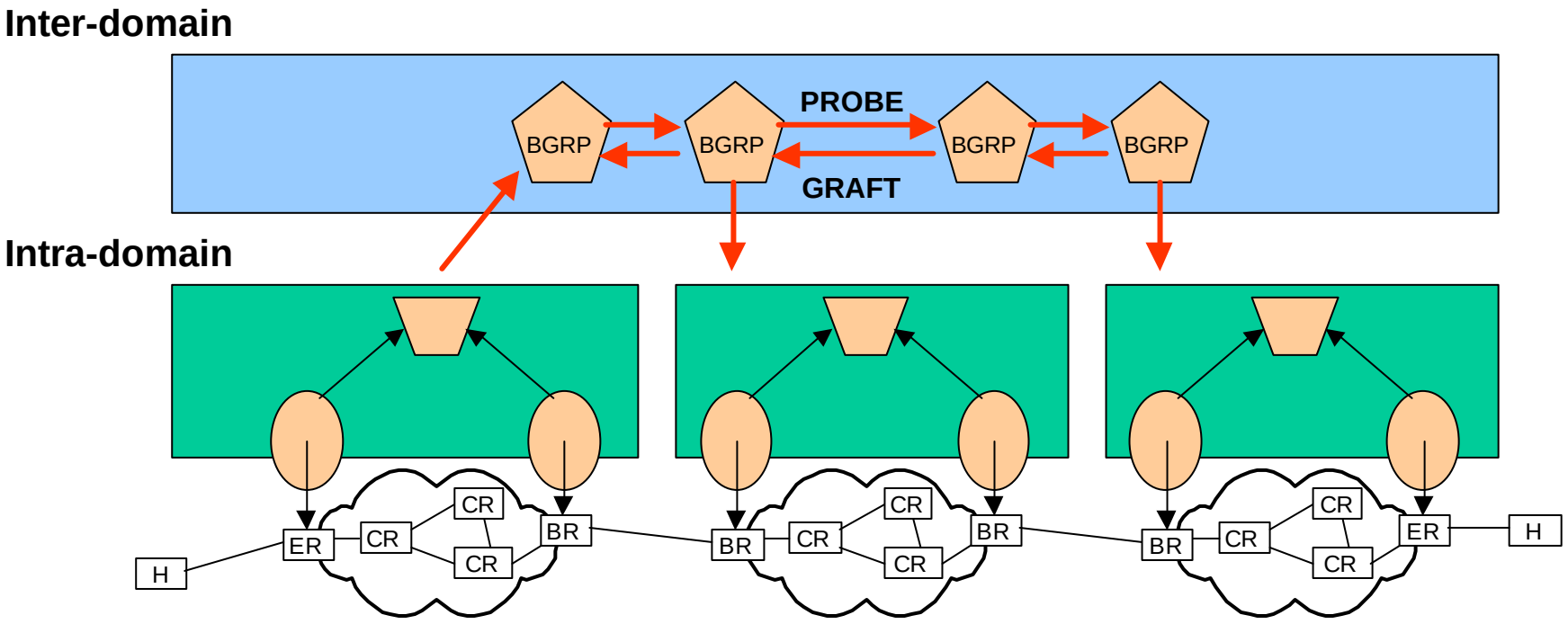
  Signalling messages (Quiet Grafting Mechanism)

Sink Trees

- Objective: state information scalability
- The sink tree is the logical representation of BGP routing
- The root of the tree is the destination domain
- Traffic is aggregated along a sink tree per destination domain
- Reservations can be aggregated as well !



BGRP+ Signalling Architecture



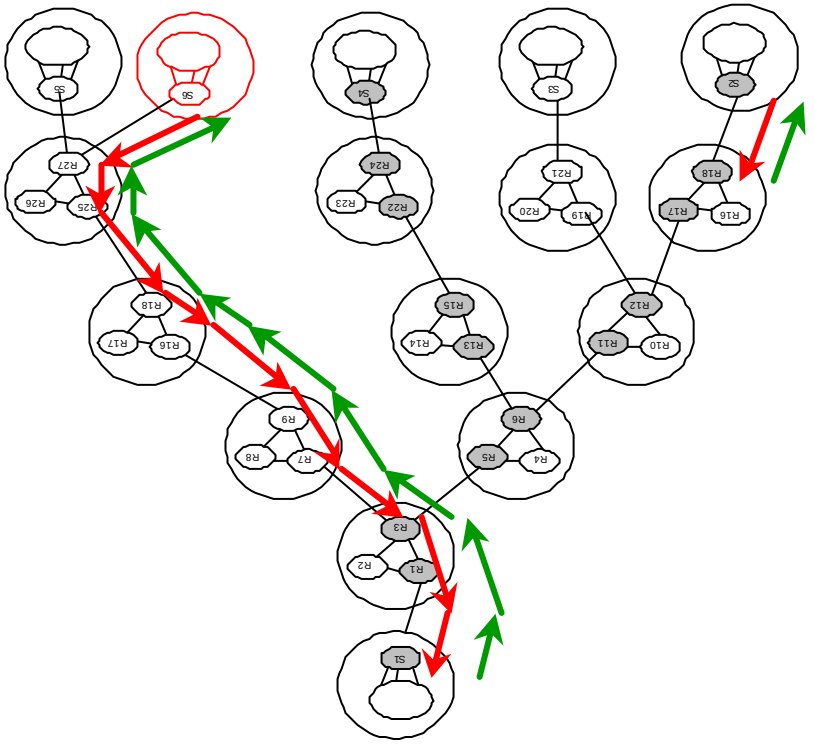
Quiet Grafting Mechanism

Objective: signalling scalability

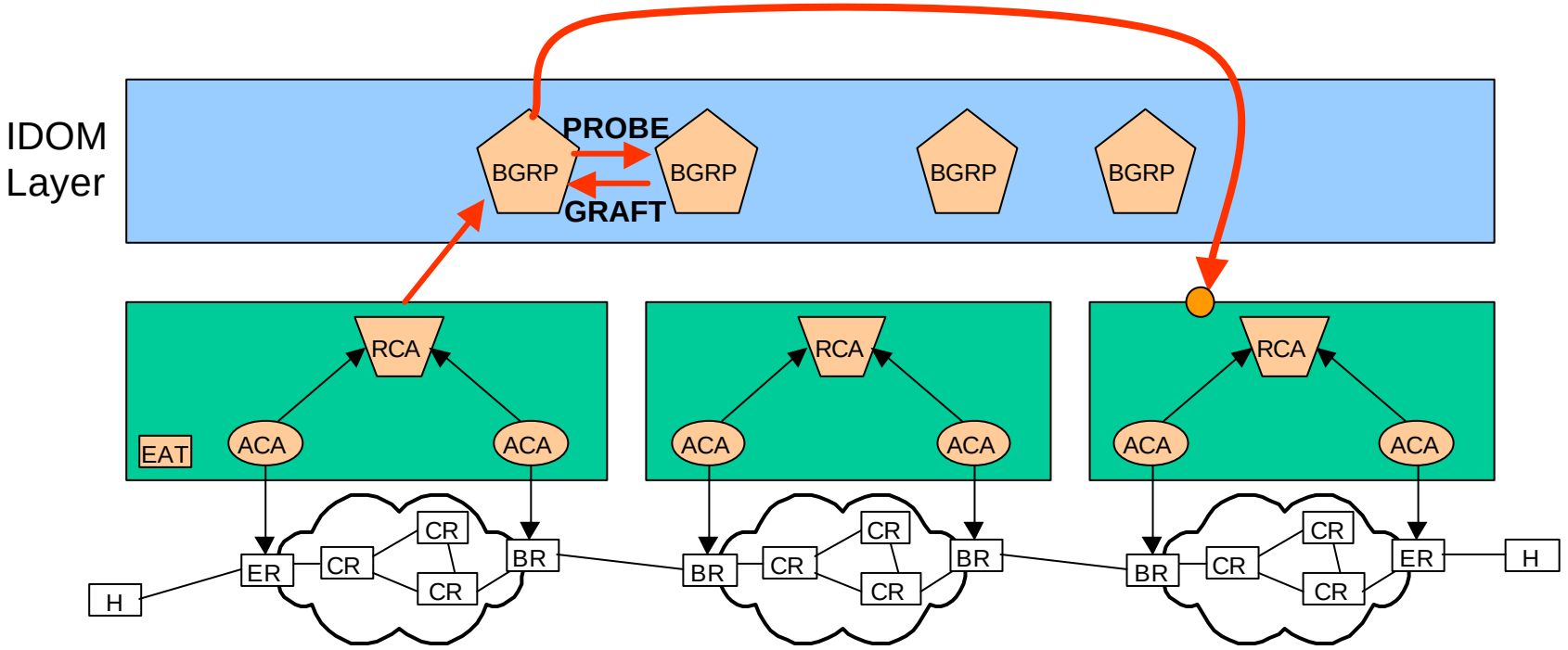
Reduce the number of PROBEs and GRAFTs

Needed components:

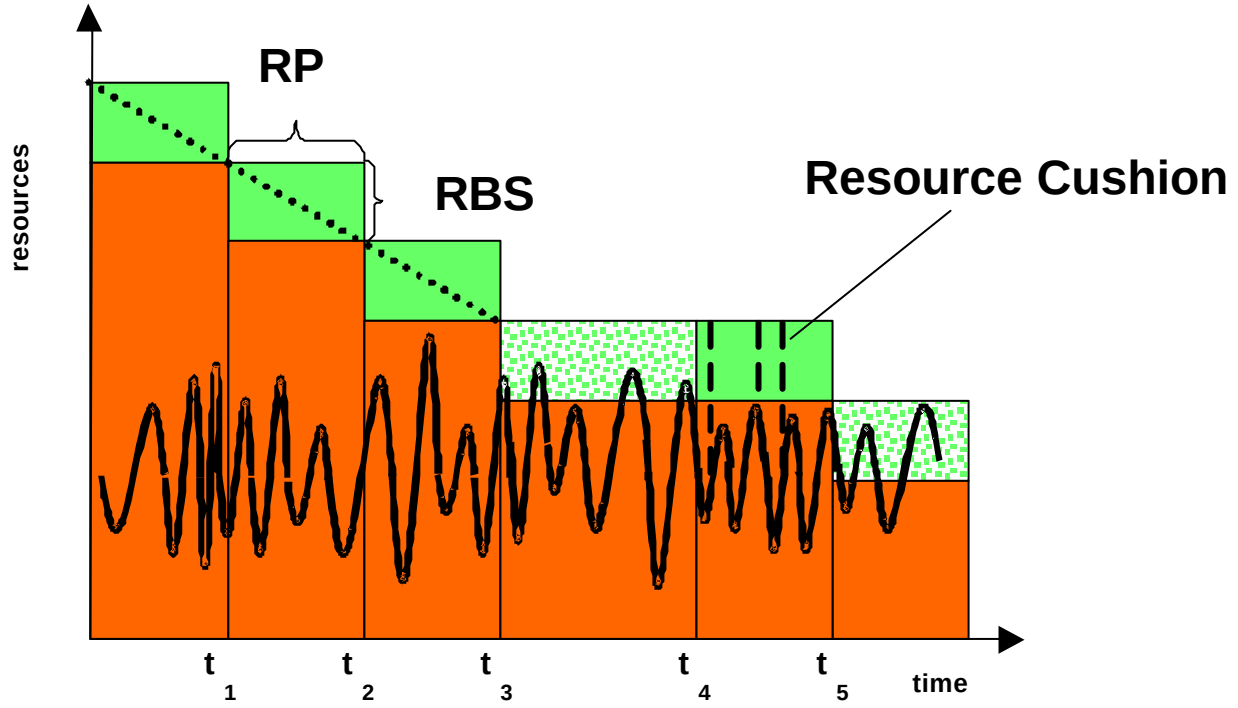
- **Sink tree identification**
  - AS, BR id
  - NLRI
- **Reservation in the last domain**



Signalling in the last Domain



Enabling Quiet Grafting Mechanism: Delayed Resource Release & Resource Cushions

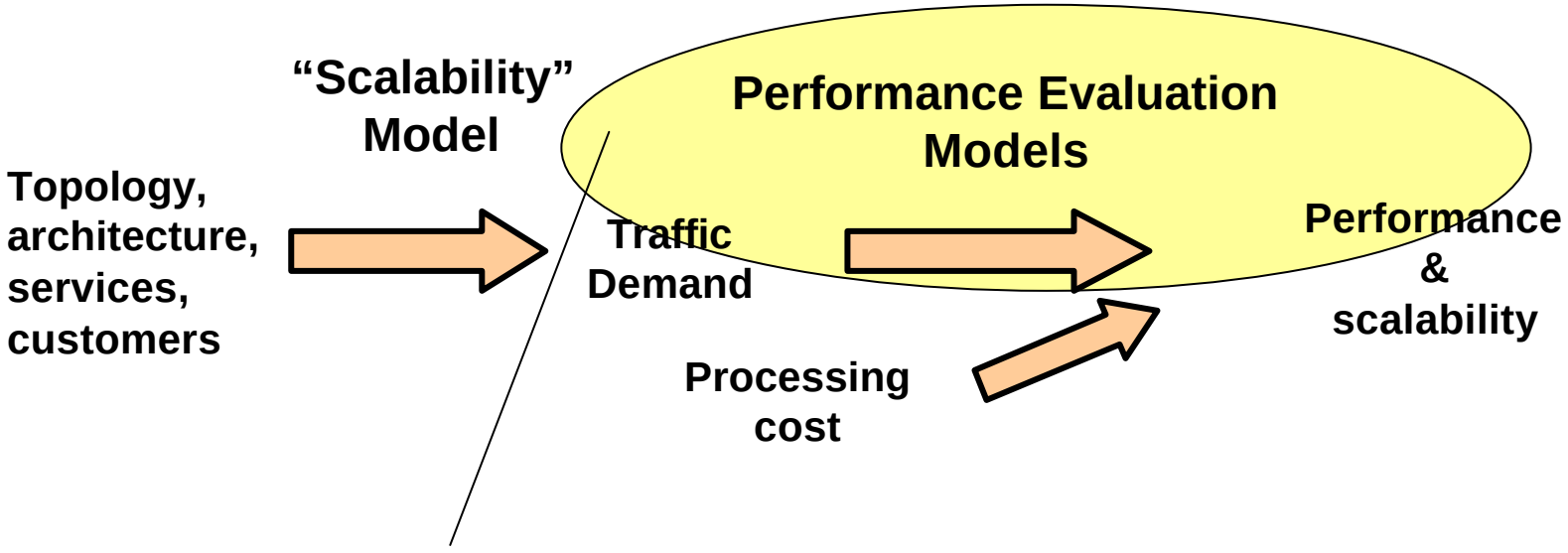


Two parameters:

- **RP Retain Period**
- **RBS Resource Block Size**

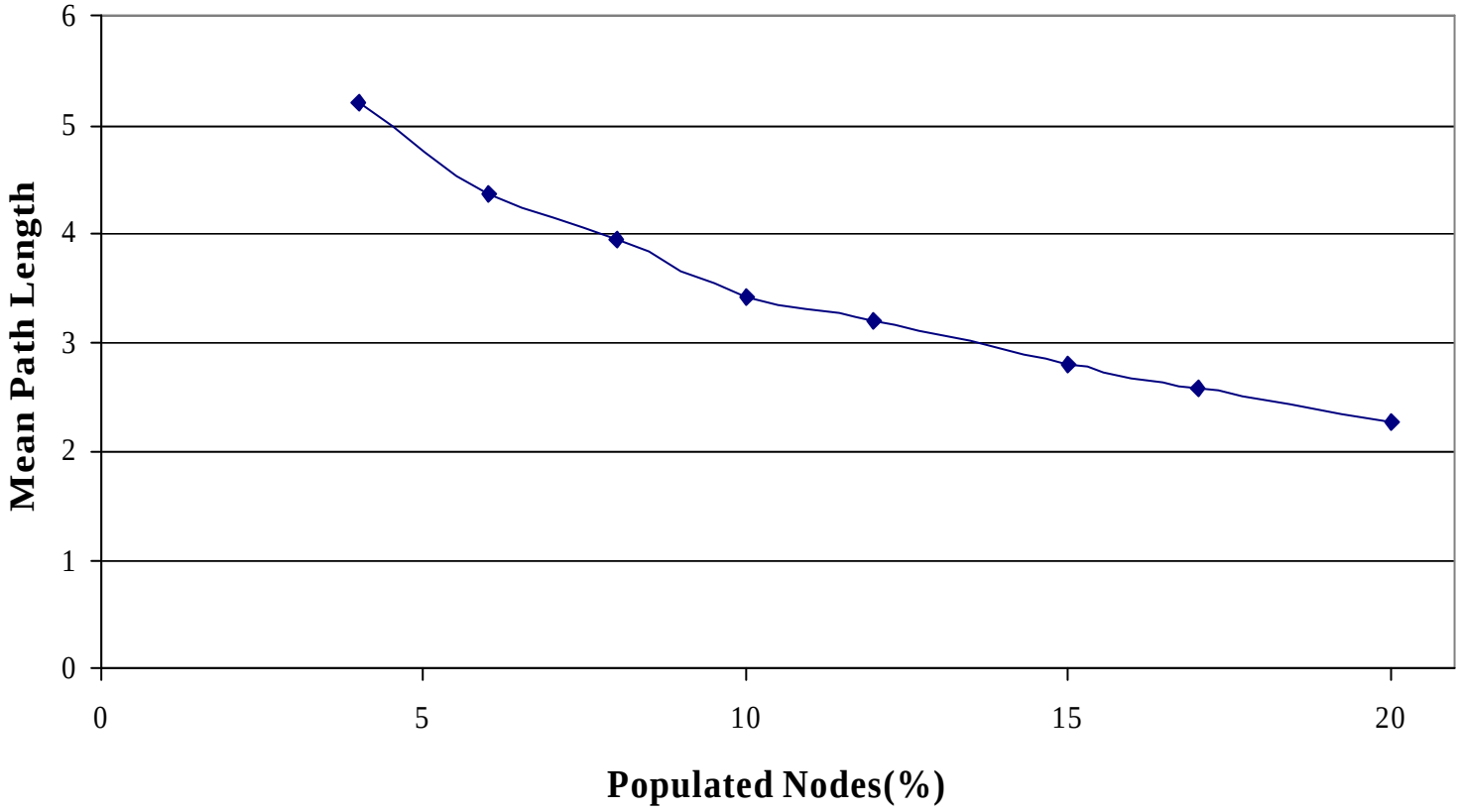
BGRP+ : Evaluating Scalability

- The scalability evaluation process is conceptually similar to the intra-domain case



- Quiet grafting point
- Effectiveness of Delayed Resource Release

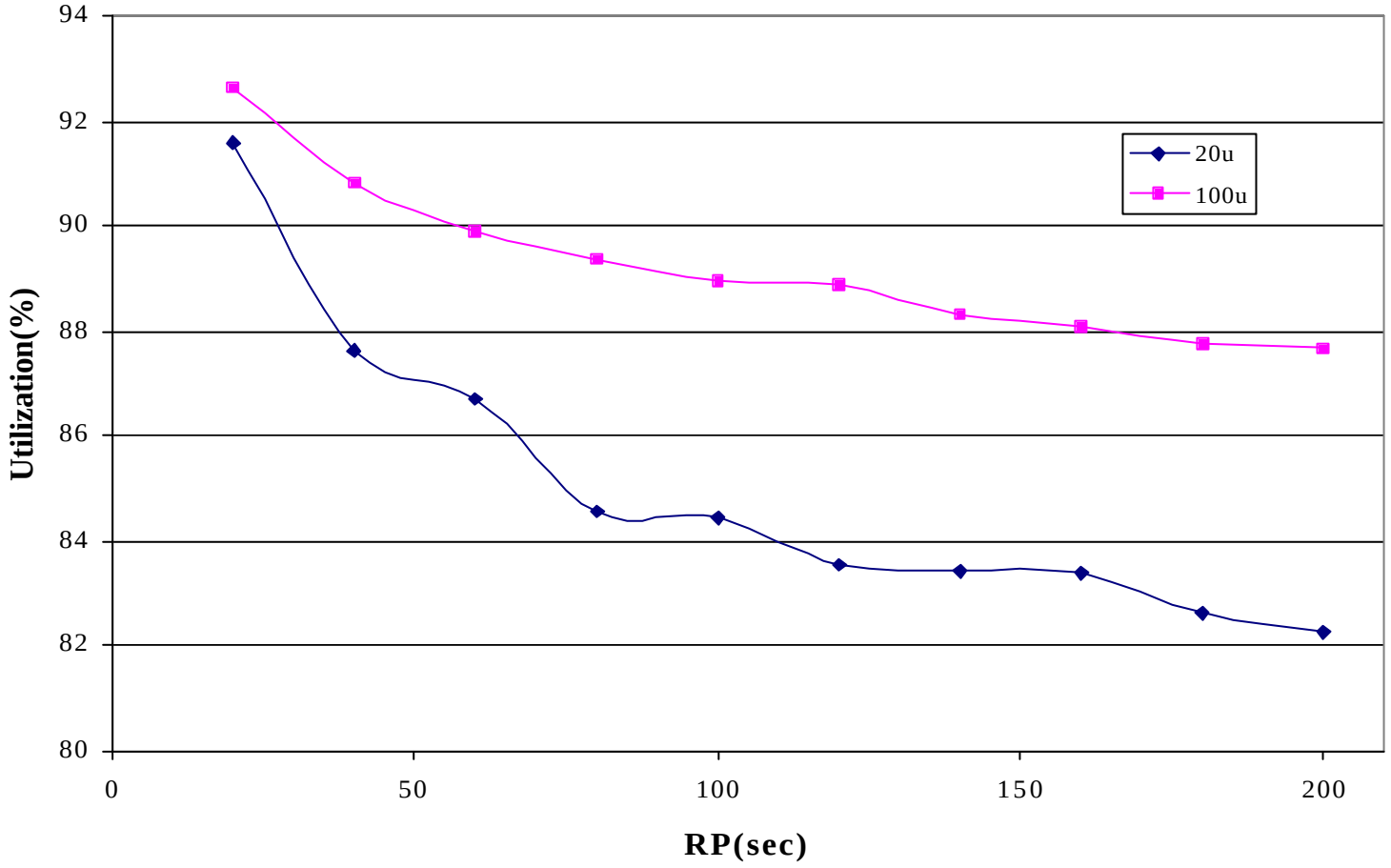
Performance study: Quiet Grafting Point Mean Path Length (Sink Tree with D=10)



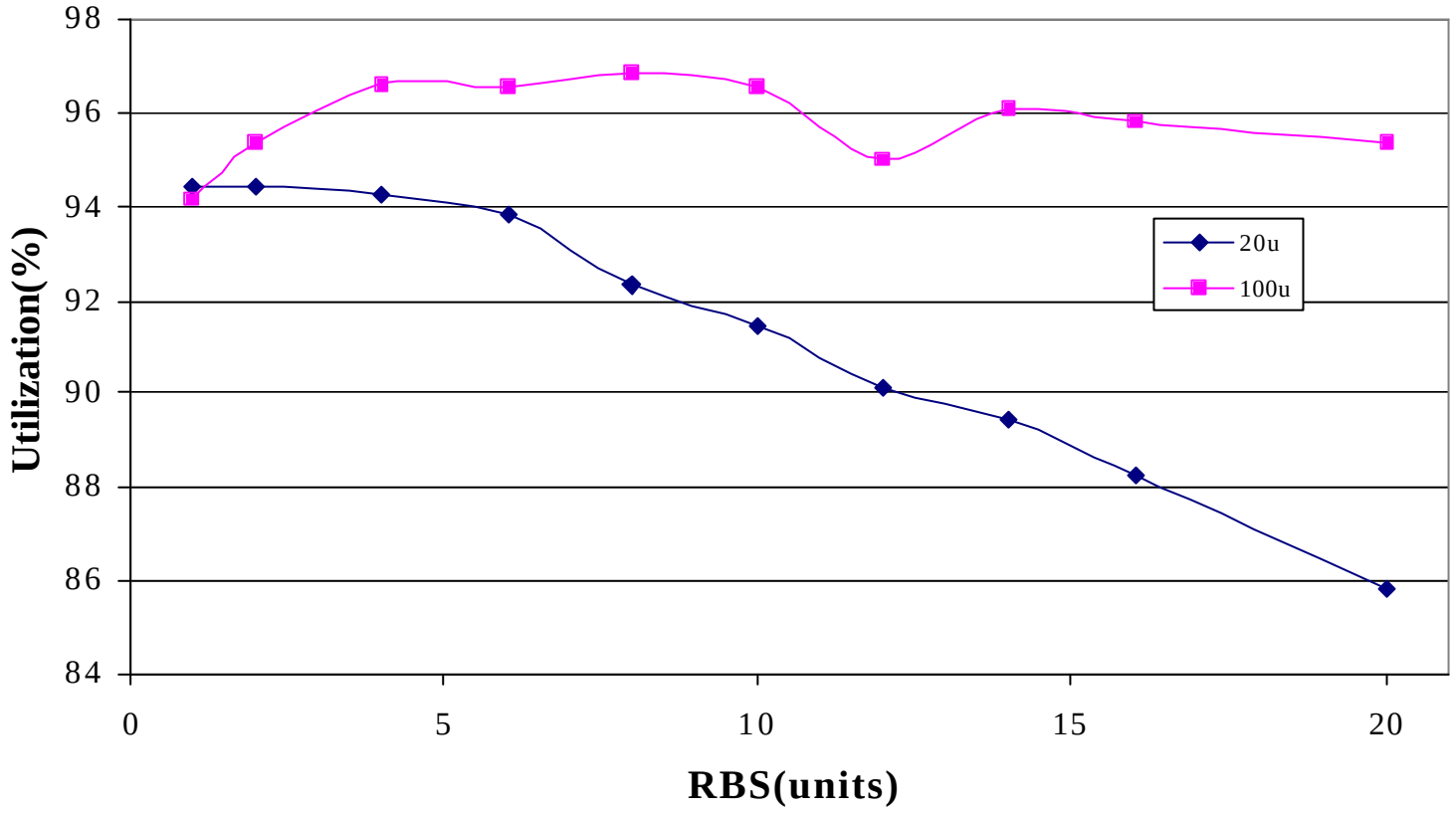
Performance Study: Effectiveness of QGM using Delayed Resource Release

- **Sink tree with depth 4**
- **Resource Requests injected from the leaves of the sink tree**
  - Traffic generators with exponential distributed inter-arrival time and holding time - Each request has fixed size: 1 unit of BW
- **Two different scenarios:**
  - Variation of the Release Block Size
  - Variation of Retain Period
- **Two load conditions:**
  - Mean offered load = 20 u
  - Mean offered load = 100 u
- **Results**
  - Utilization of the overall resources of the sink tree
  - Signaling load
- **Trade-off**
  - Higher RP yields lower Utilization but lower Signaling load
- **Load dependent**
  - Higher load gives better results

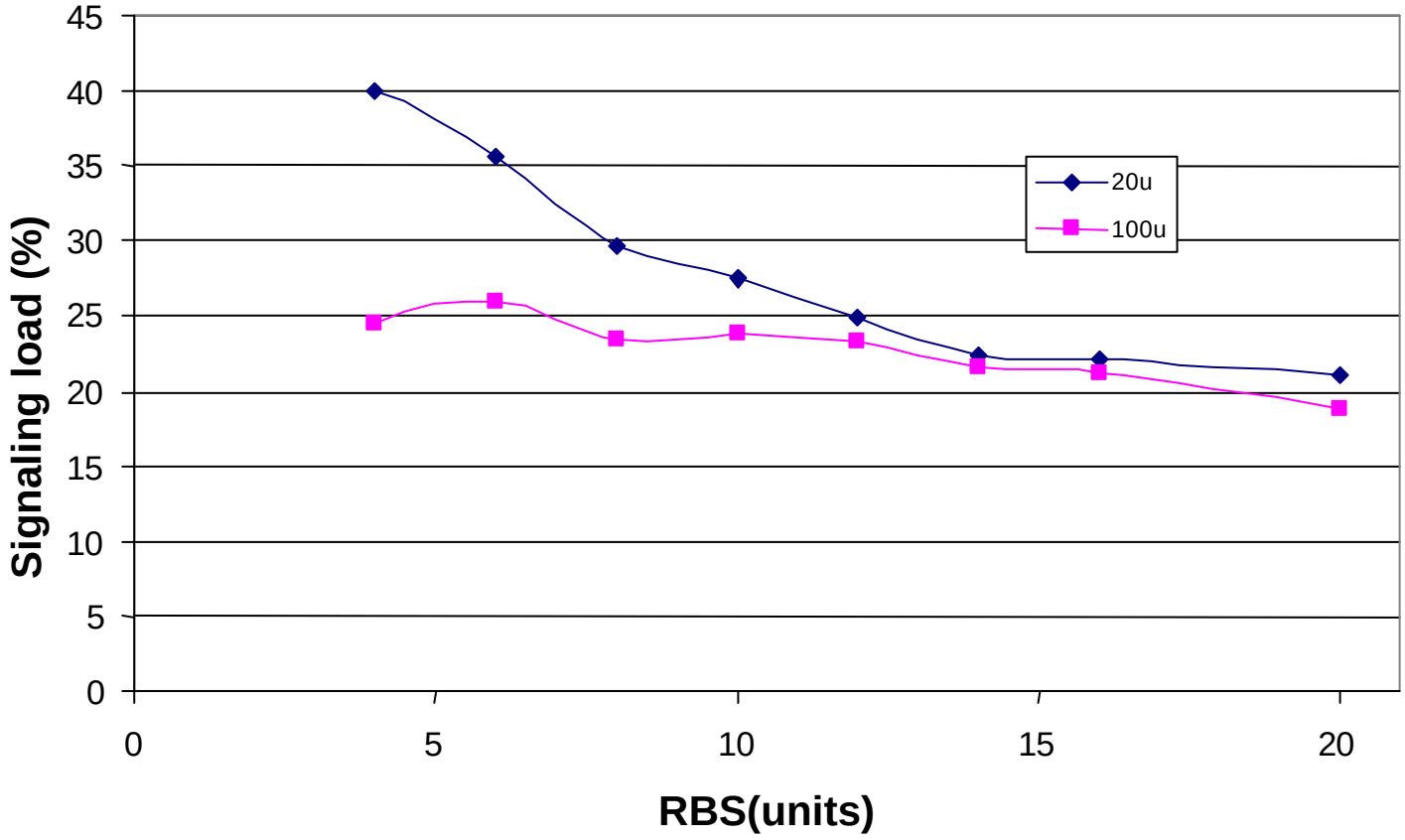
Utilisation vs. Retain Period (at fixed RBS=1 unit)



Utilisation vs. Resource Block Size (at fixed RP=10sec)



Signaling load vs. Resource Block Size (at fixed RP=10sec)



Discussion on Current Inter-domain Performance Results

- **Proof of concept**
  - Delayed resource release can dramatically reduce the signalling load
- **Delayed Resource Release Algorithm performance**
  - RP has a profound effect on the performance of the algorithm
  - RBS does not effect performance under high load conditions
  - In general, algorithm more sensitive to RP than RBS
- **Especially effective in lively branches of the sink tree**
  - Best results when number of incoming requests is high
- **Optimal parameter sets will vary at different places in the sink tree**
  - Short RP and small RBS near the leafs
  - Longer RP and larger RBS near the sink

Conclusions

- Dynamic IP QoS is scalable in the intra-domain case
- The AQUILA architecture is a very good proof
- As for the inter-domain case … after first simulation results we are confident, it is, and we are still working on it

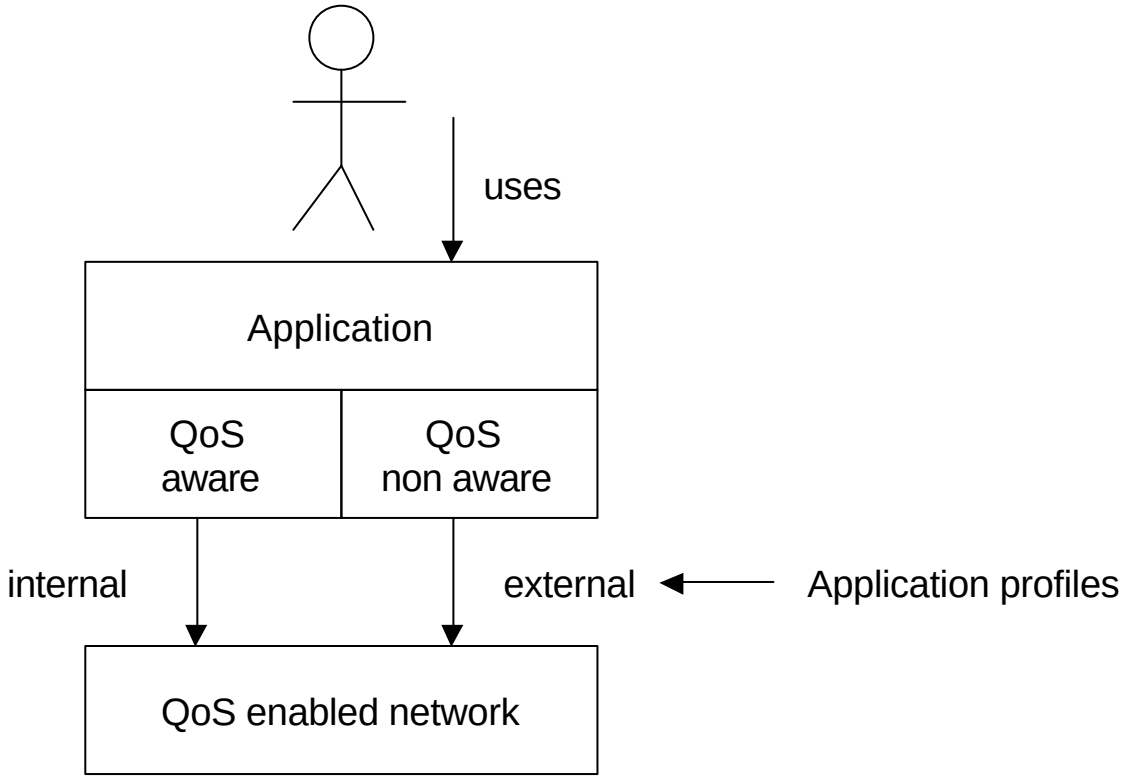
Application Profiles

Outline

- Introduction and approach
- Application profiles - syntax
- Application profiles - instance
- Reservation example
- Conclusion

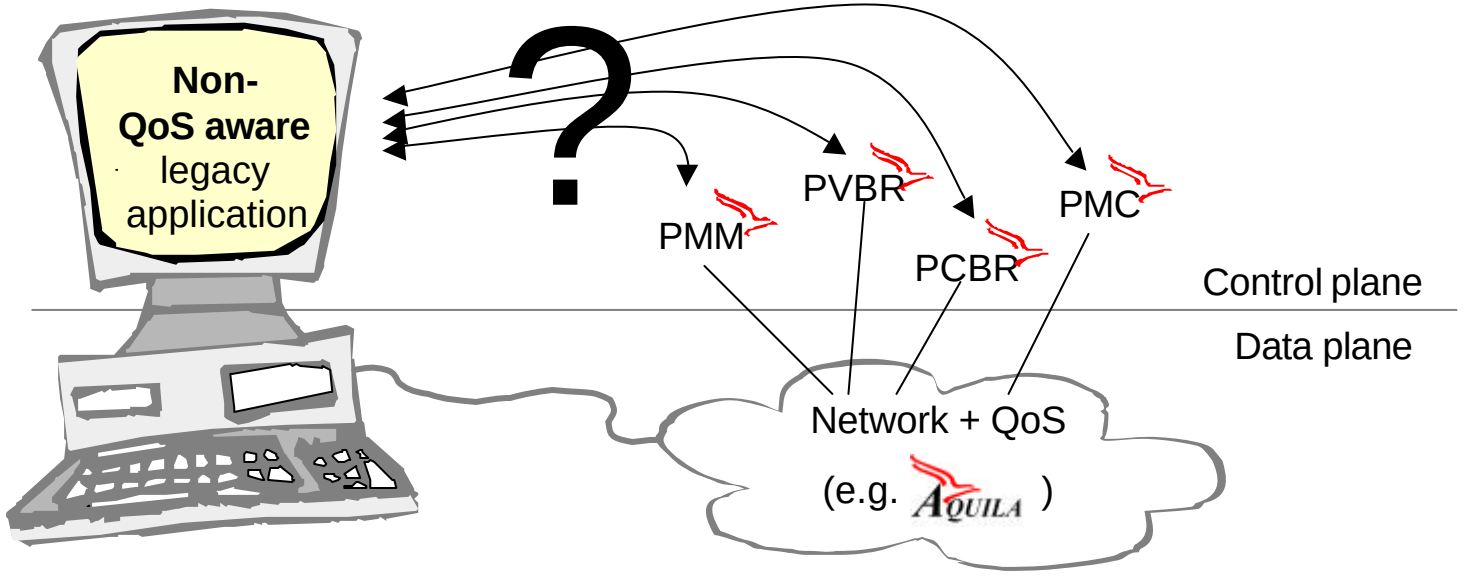
Motivation

- **Mapping of end-user QoS into network QoS**



Initial Situation

- **Non-QoS aware legacy application**
  - cannot request, present/offer QoS
- **QoS aware infrastructure e.g. AQUILA**

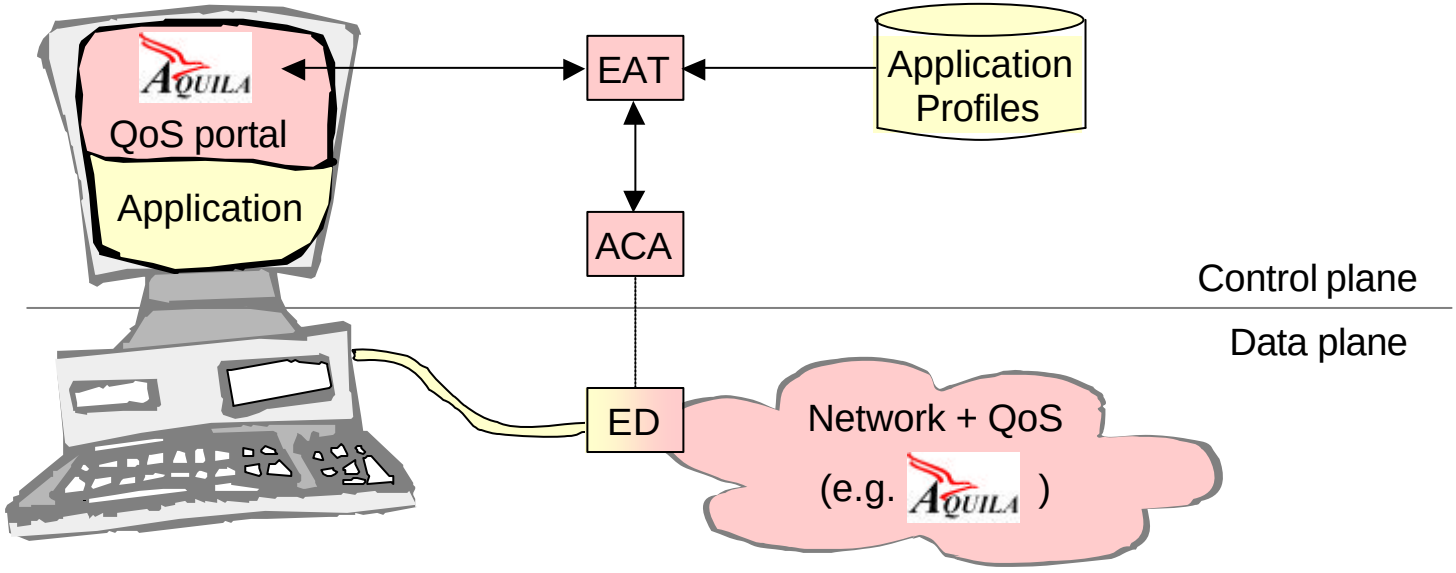


Objectives

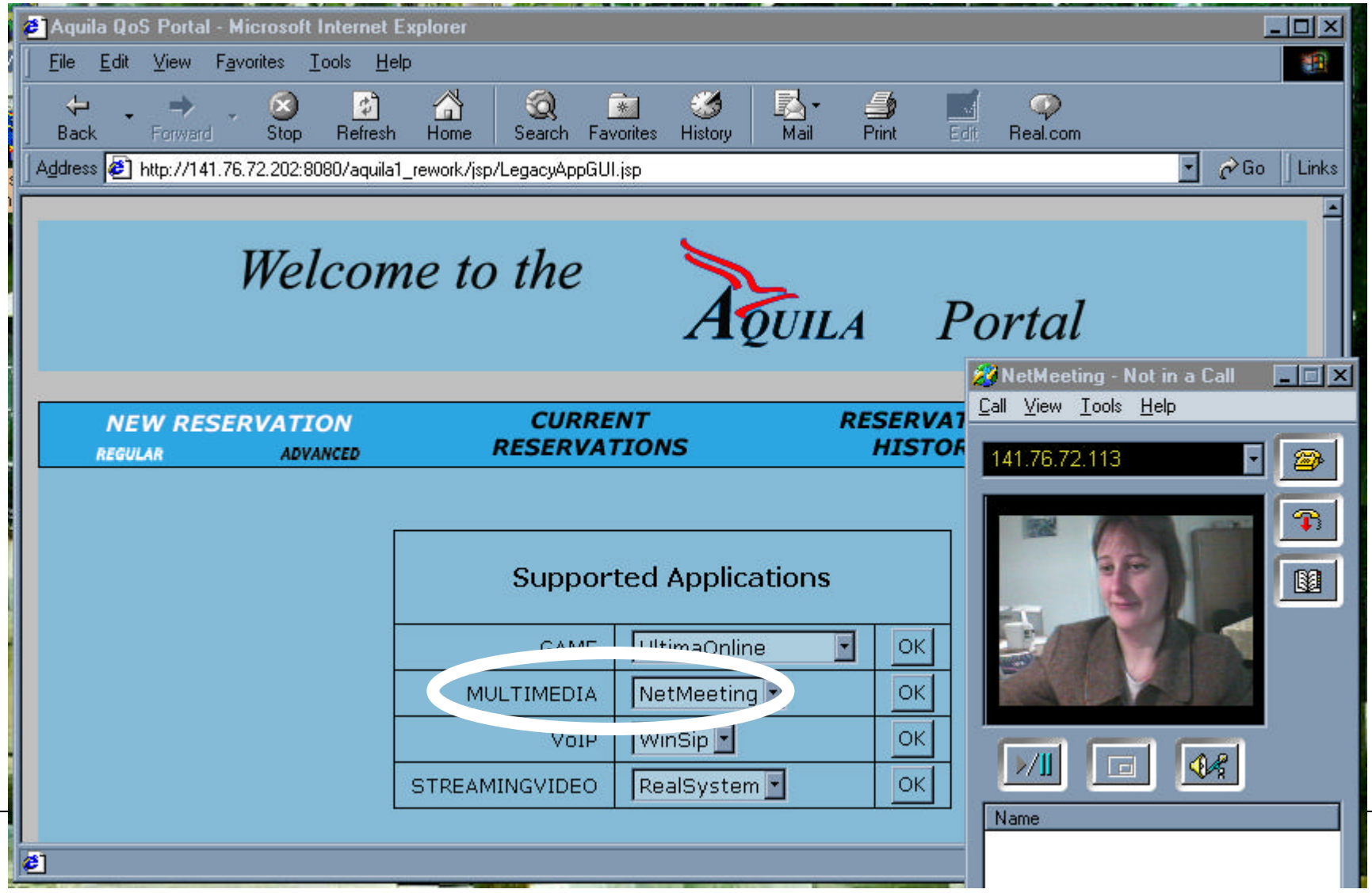
- **Supply non-QoS aware legacy applications with QoS**
  - Keep unchanged the legacy applications
  - No re-/new programming!
  - Present QoS to end-user
- **Specification of QoS at**
  - End-user level
  - Application level
    - data plane
    - control plane
  - Network level

AQUILA Solution at End-user Level

- **QoS is offered at the beginning of application runtime**
- **"QoS portal"**
  - supports QoS supply
  - presents QoS metaphors



Example - NetMeeting and Portal



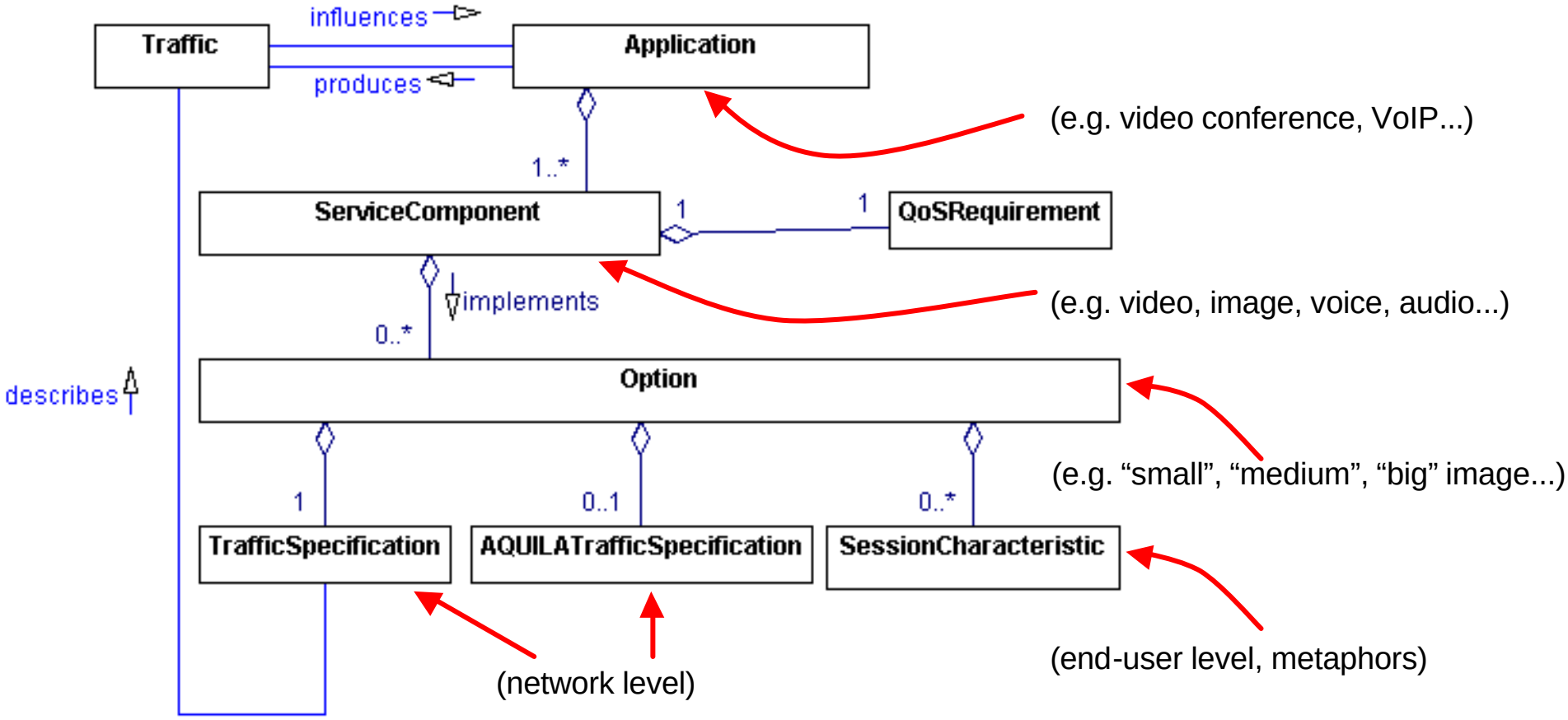
Outline

- Introduction and approach
- **Application profiles - syntax**
- Application profiles - instance
- Reservation example
- Conclusion

Application Profiles - Syntax

- **Network level**
  - how to describe the AQUILA QoS request - implementation dependent
  - how to describe the QoS expectations / requirements
  - how to describe the produced traffic
- **Application level**
  - protocol used, port used... (control plane)
  - implementation issues of service components (data plane)
- **End-user level**
  - how to build metaphors → QoS portal

Application Structure

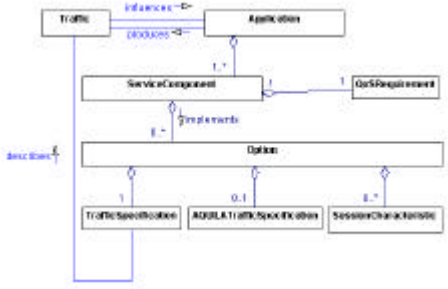


Syntax Application Profile - DTD

- **Document Type Definition for XML documents**
- **Follow application structure**

ApplicationProfile (Implementation+, protocol*)

- protocol (lowerPortNo?,upperPortNo?,isControlPort?)
- Implementation (ServiceComponent, TransportProtocol)
  - ServiceComponent (QoSRequirement, Option+)
    - » QoSRequirement (maxDelay, maxJitter, maxLoss, bw, ordering)
    - » Option (TrafficSpecification?, AQUILASpec?, SessionCharacteristic+)
      - TrafficSpecification (type+, duration, adaptivity, burstiness, packetSize, bitRate, flow)
      - AQUILASpecification (serviceID, BSP, BSS, minPU, maxPS, PR, SR)
      - SessionCharacteristic (name, semanticalGroup*)



? 0 - 1

* 0 .. N

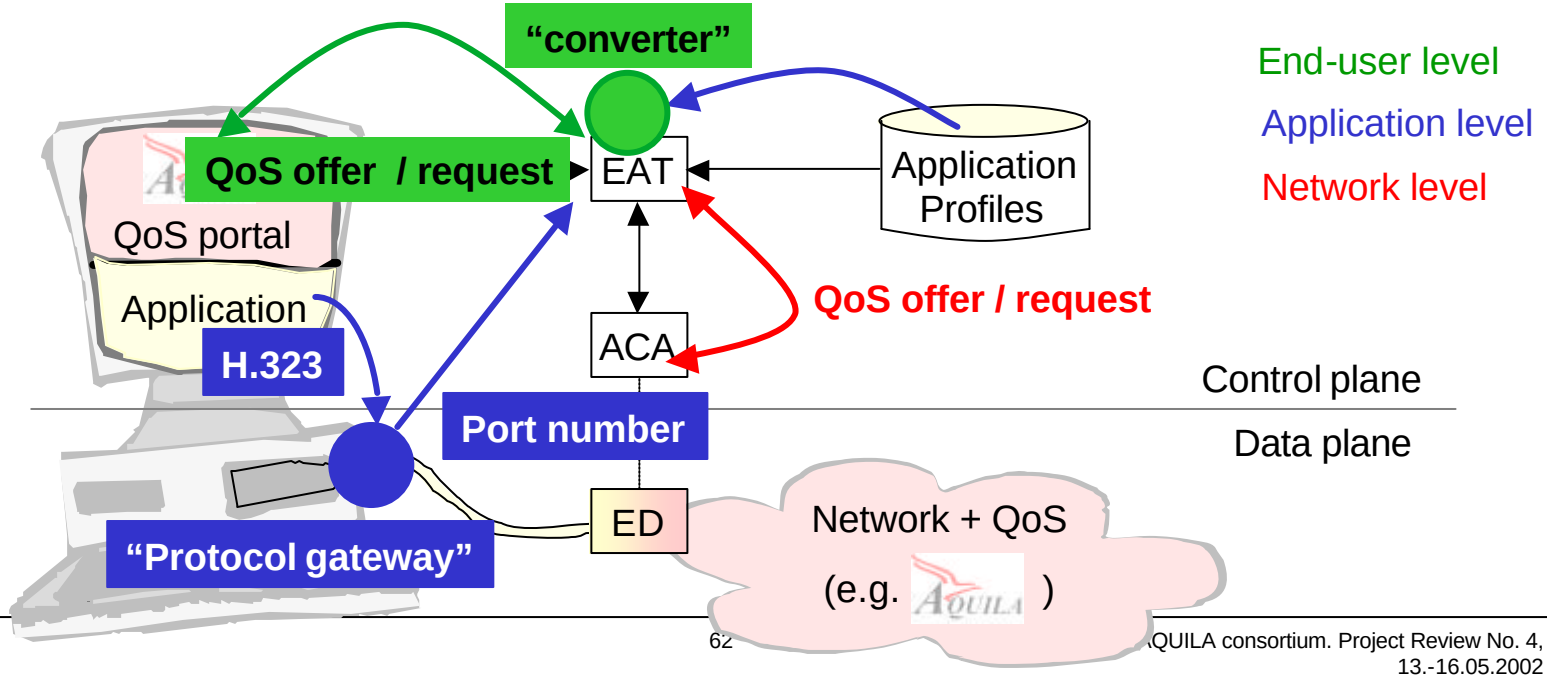
+ 1..n

Outline

- Introduction and approach
- Application profiles - syntax
- **Application profiles - instance**
- Reservation example
- Conclusion

Application Profile Instance - XML

- **For each application one profile**
  - Parameters & values for conversion
  - Low-level control plane information
  - End-user metaphors to produce the QoS portal



Application Level - Control Plane

```
<ApplicationProfile   build="4.4.3388" type="MULTIMEDIA" version="3.01"
                      name="NetMeeting" scope="xdirectional">
        <Implementation>
                ...
        </Implementation>
       <protocol name="H323">
                <isControlPort value="true">1720</isControlPort>
        </protocol>
        <protocol name="RTP">
                <isControlPort value="false"></isControlPort>
        </protocol>
</ApplicationProfile>
```

Application Level - Data Plane

```
<Implementation>
        <ServiceComponent file="NetMeeting_3_01_Video_v2.xml">
            <name>NetMeeting_3_01_Video_v2</name>
        </ServiceComponent>
        <TransportProtocol name="TCP"/>
 </Implementation>
 <Implementation>
        <ServiceComponent file="NetMeeting_3_01_Speech_v2.xml">
             <name>NetMeeting_3_01_Speech_v2</name>
        </ServiceComponent>
        <TransportProtocol name="TCP"/>
 </Implementation>
```

Network Level - AQUILA

```
<ServiceComponentProfile name="NetMeeting_3_01_Video_v2" ServiceComponent="VIDEO">
    <QoSRequirement>
        <maxDelay requirement="high" weight="1" unit="ms">1200</maxDelay>
        <maxJitter weight="3" unit="ms" requirement="low">120</maxJitter>
        <maxLoss weight="5" unit="percent" requirement="medium">10</maxLoss>
        <bwGuarantee weight="8" unit="percent" requirement="high">-1</bwGuarantee>
        <ordering weight="8" requirement="true" />
    </QoSRequirement>
    <Option description="video very low quality scenario" ...> …
        <AQUILASpecification>
            <serviceID value="PVBR" />
            <BSP unit="bytes">2000</BSP>
            <BSS unit="bytes">5120</BSS>
            <minPU unit="bytes">60</minPU>
            <maxPS unit="bytes">1500</maxPS>
            <PR unit="bit/s">160000</PR>
            <SR unit="bit/s">75000</SR>
        </AQUILASpecification>
    </Option>
```

End-user Level - Metaphors

```
<Option  description="video very low quality scenario" ...>
  <SessionCharacteristic>
        <name>Picture size</name>
        <semanticalGroup     type="UserFriendly"          language="en">
              <description>picture size</description>
              <qualifier>very small</qualifier>
        </semanticalGroup>
        <semanticalGroup     type="UserFriendly"          language="fr">
              <description>taille de l'image</description>
              <qualifier>très petite</qualifier>
        </semanticalGroup>
        <semanticalGroup     type="UserFriendly"          language="de">
              <description>Bildgröße</description>
              <qualifier>sehr klein</qualifier>
        </semanticalGroup>
  </SessionCharacteristic>
...
```

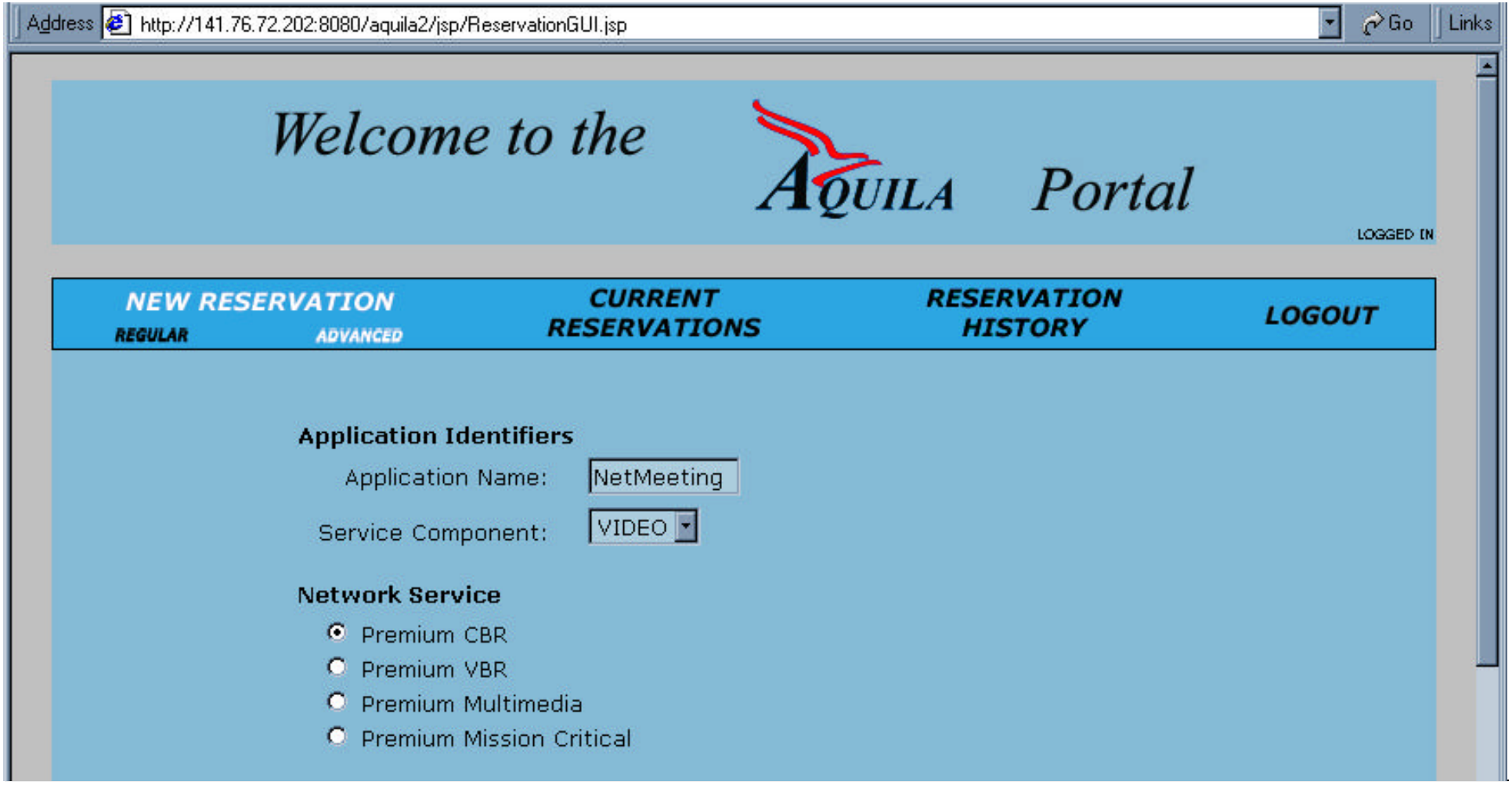
Outline

- Introduction and approach
- Application profiles - syntax
- Application profiles - instance
- **Reservation example**
- Conclusion

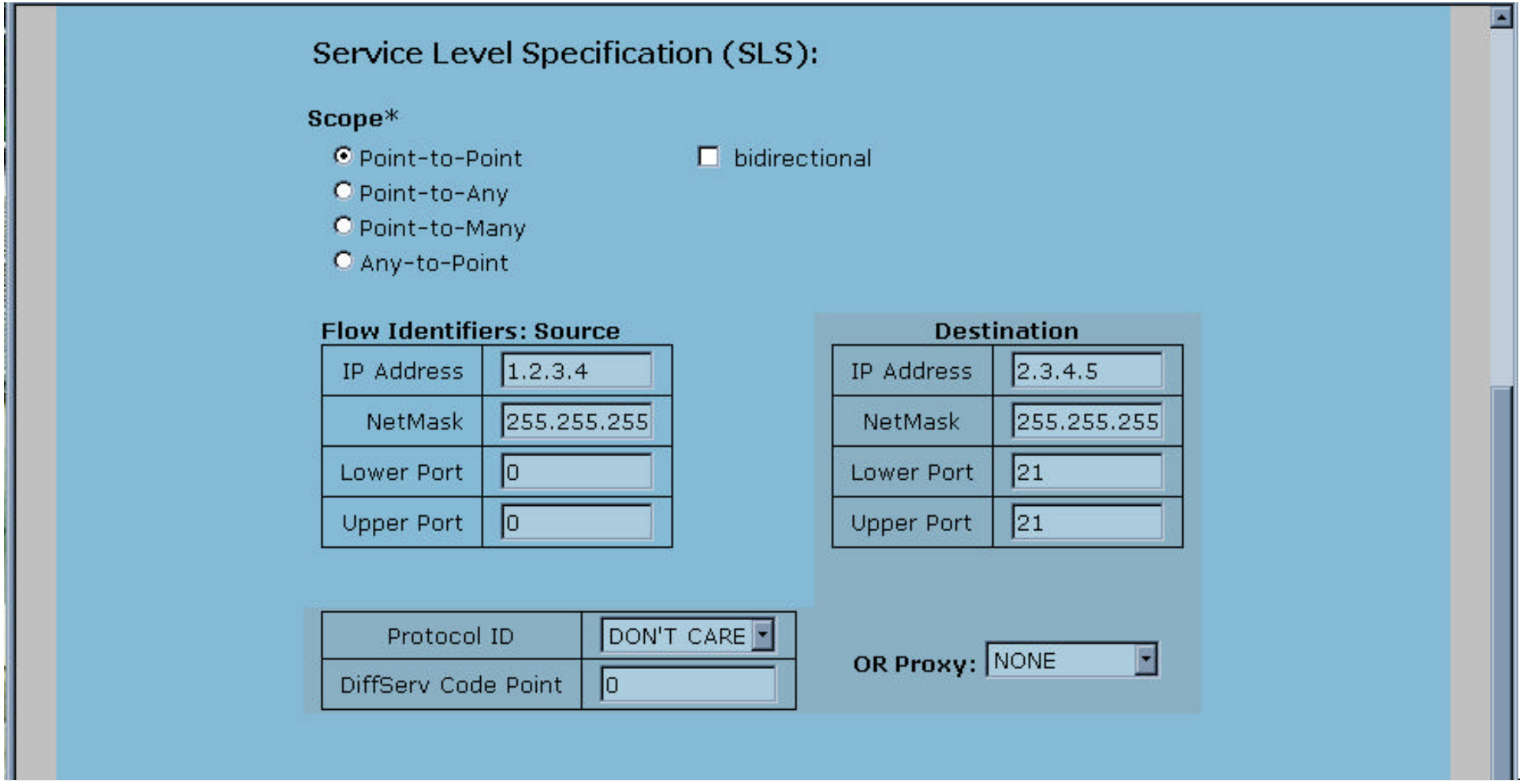
Example without Application Profile

- **Low-level management tool**
  - Network oriented
- **Difficult to handle**
- **Only for experienced users**
- **Error prone**

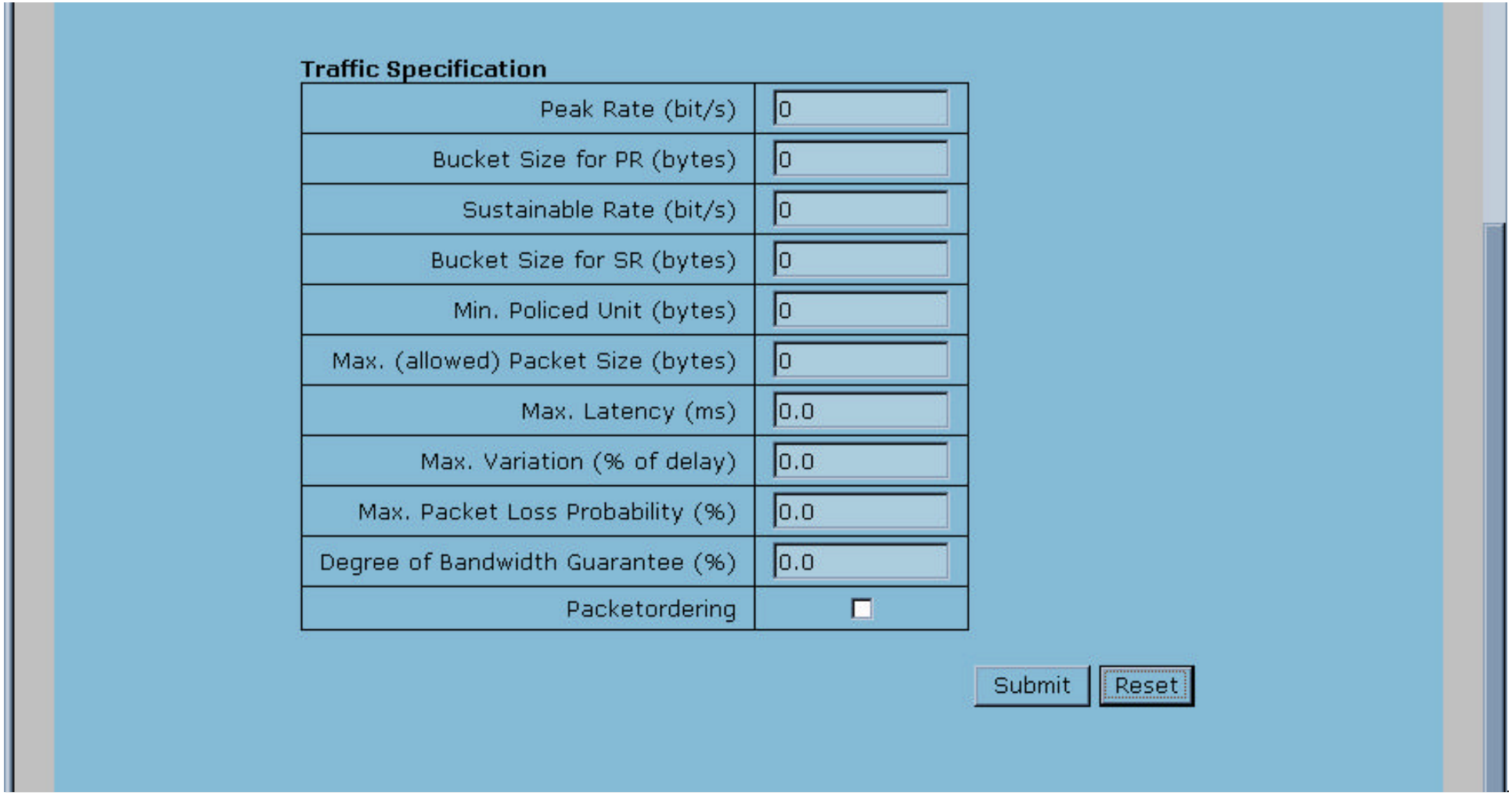
Reservation without Profiles (1)



Reservation without Profiles (2)



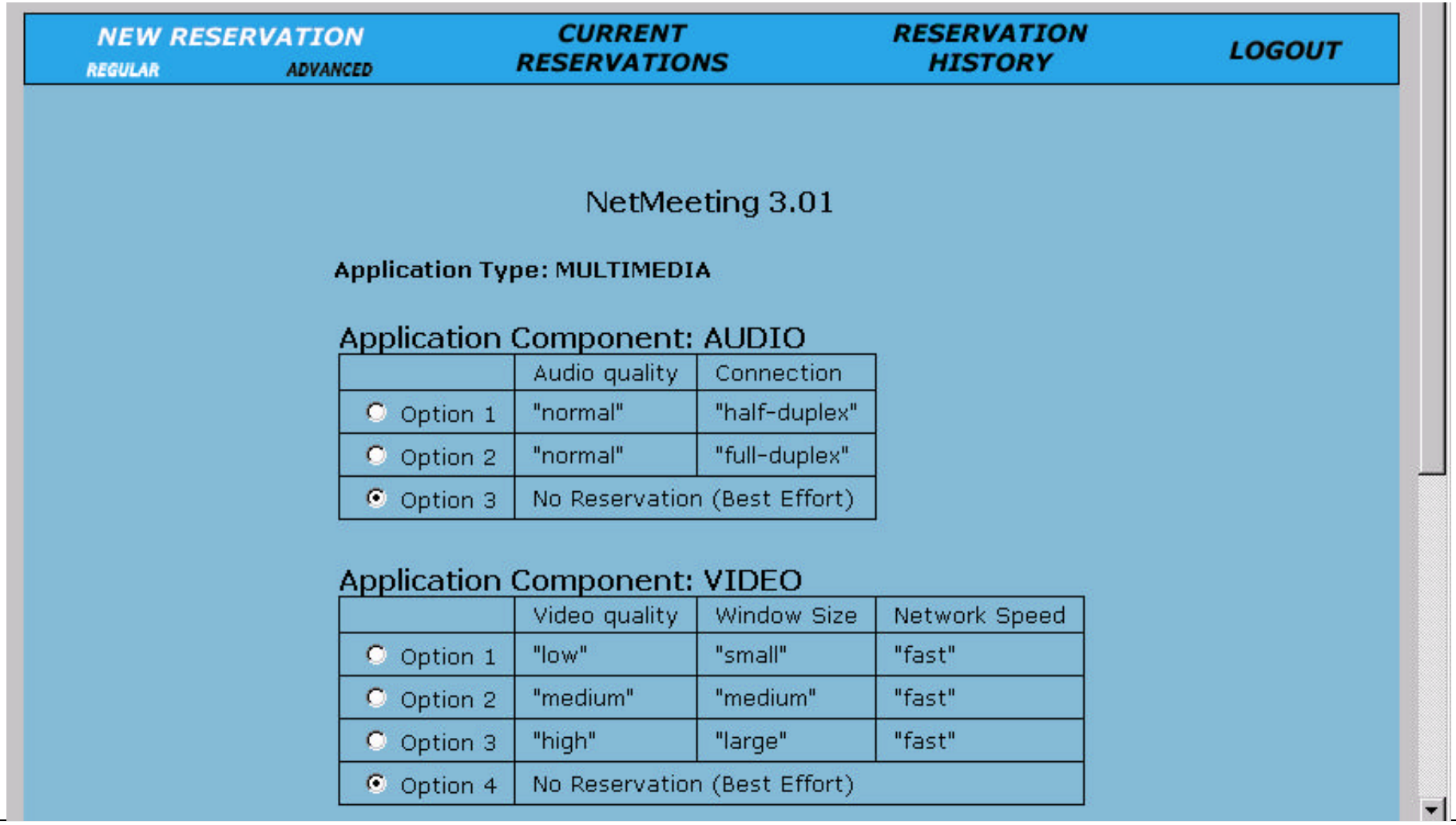
Reservation without Profiles (3)



Example with Application Profile

- **High-level provisioning tool**
  - End-user oriented
- **Easy to handle**
- **Even for unexperienced users**
- **Self-explaining and concise**

Reservation with Profiles



Outline

- Introduction and approach
- Application profiles - syntax
- Application profiles - instance
- Reservation example
- Conclusion

Conclusion

- Mechanism for supplying non-QoS aware applications with QoS
- Toolkit for building QoS aware applications
- Application profiles define a generic syntax for describing applications
- Building a repository
  - RealPlayer
  - NetMeeting
  - SIP application
  - …
- http://www.ist-aquila.org

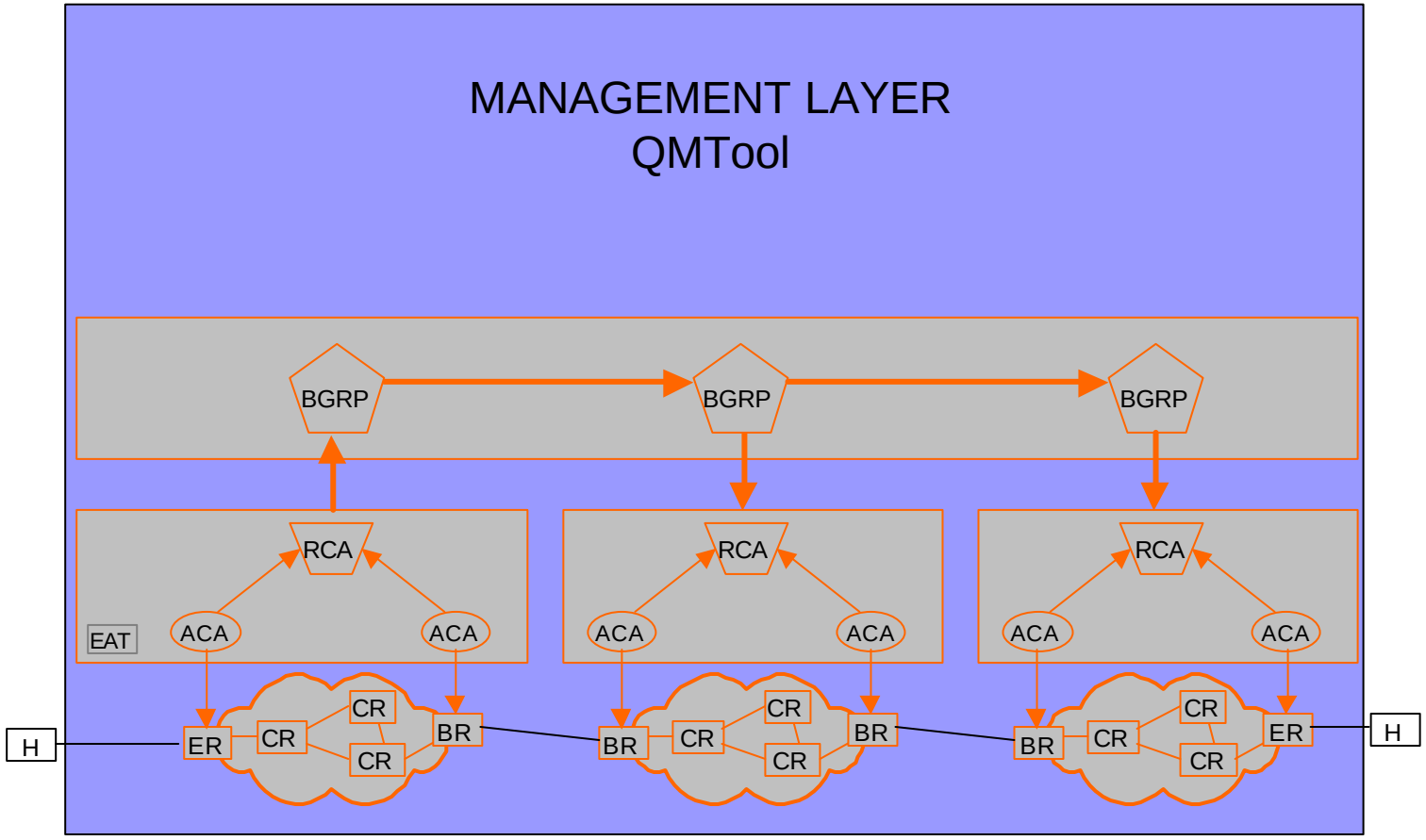
Quality Of Service Management Tool

QMTool

Outline

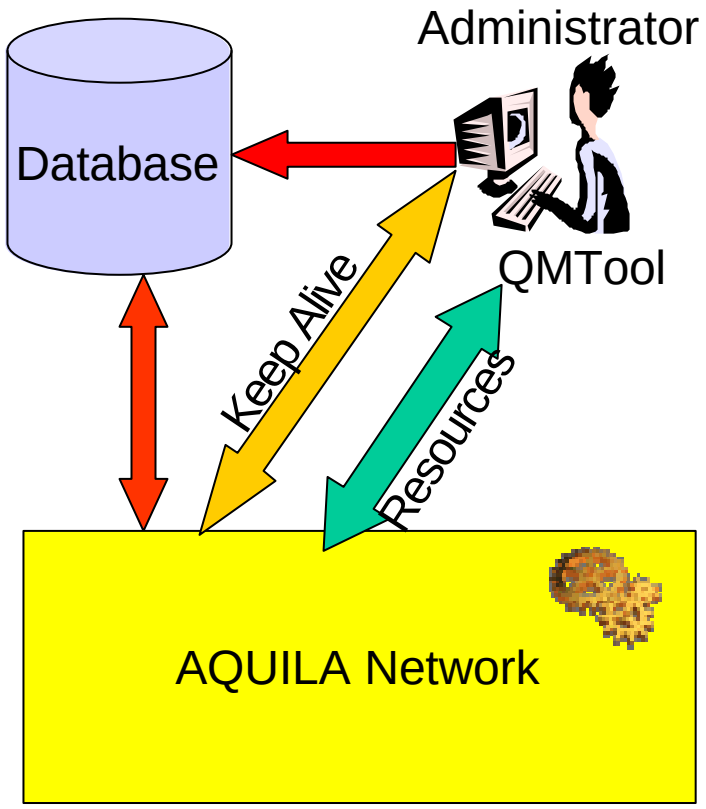
- QMTool approach
- QMTool functionality
  - Design
  - Configuration
  - Failure detection
  - Monitoring

AQUILA Network – Where does QMTool fit?



General Description

- **Configuration of the network**
  - Storage of configuration information into the database
  - Retrieval of this information by the AQUILA network components
  - Start up of the AQUILA Network
- **Failure detection of the network components**
  - Keep Alive mechanism for detecting the failure of a component
- **Monitoring of the resource utilisation**
  - Monitoring of the Resource Pool's and BGRP Agent's resources



Outline

- QMTool approach
- QMTool functionality
  - Design
  - Configuration
  - Failure detection
  - Monitoring

How does this Work?

- **QMTool provides a GUI interface for the design of the AQUILA network**

  which means:

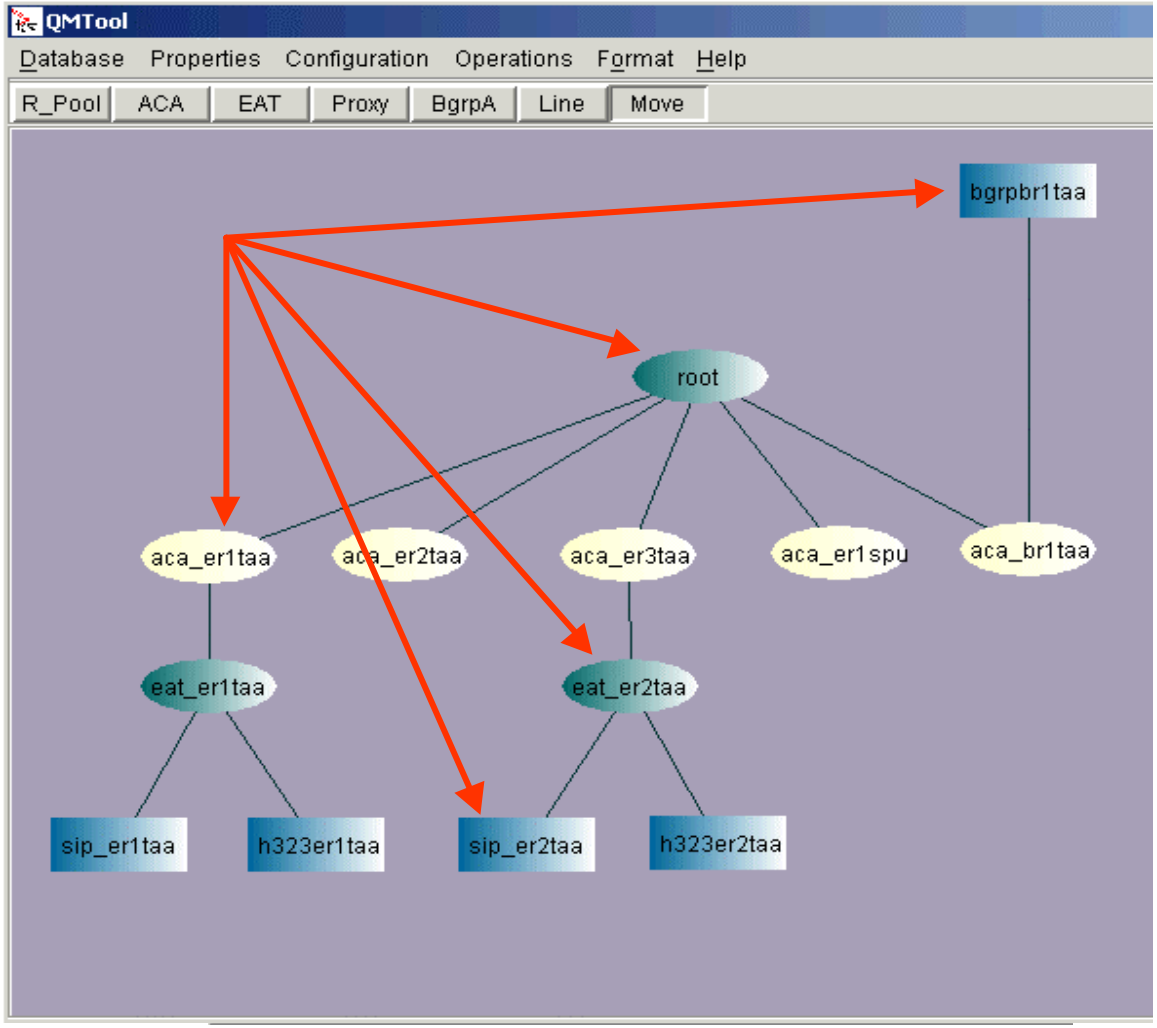
  - Design of the Resource Control Layer, i.e. Resource Pools, ACAs, EATs, Proxies
  - Design of the Inter-Domain Layer, i.e. BGRP Agents
  - Design of the connections between these components

  and if I connect two components that are not related?

  - QMTool will not let you!
  - It is aware of the valid relations between components, for example Resource Pool – ACA, ACA – Proxy

Design Demonstration

- Toolbar with all the components for design
- “Import” option for creating the network architecture by reading the information from the database
- Design of the AQUILA network



Outline

- QMTool approach
- QMTool functionality
  - Design
  - Configuration
  - Failure detection
  - Monitoring

How do I Configure the Network?

- **QMTool is integrated with an XML Editor, the XMLOperator**

  which means:

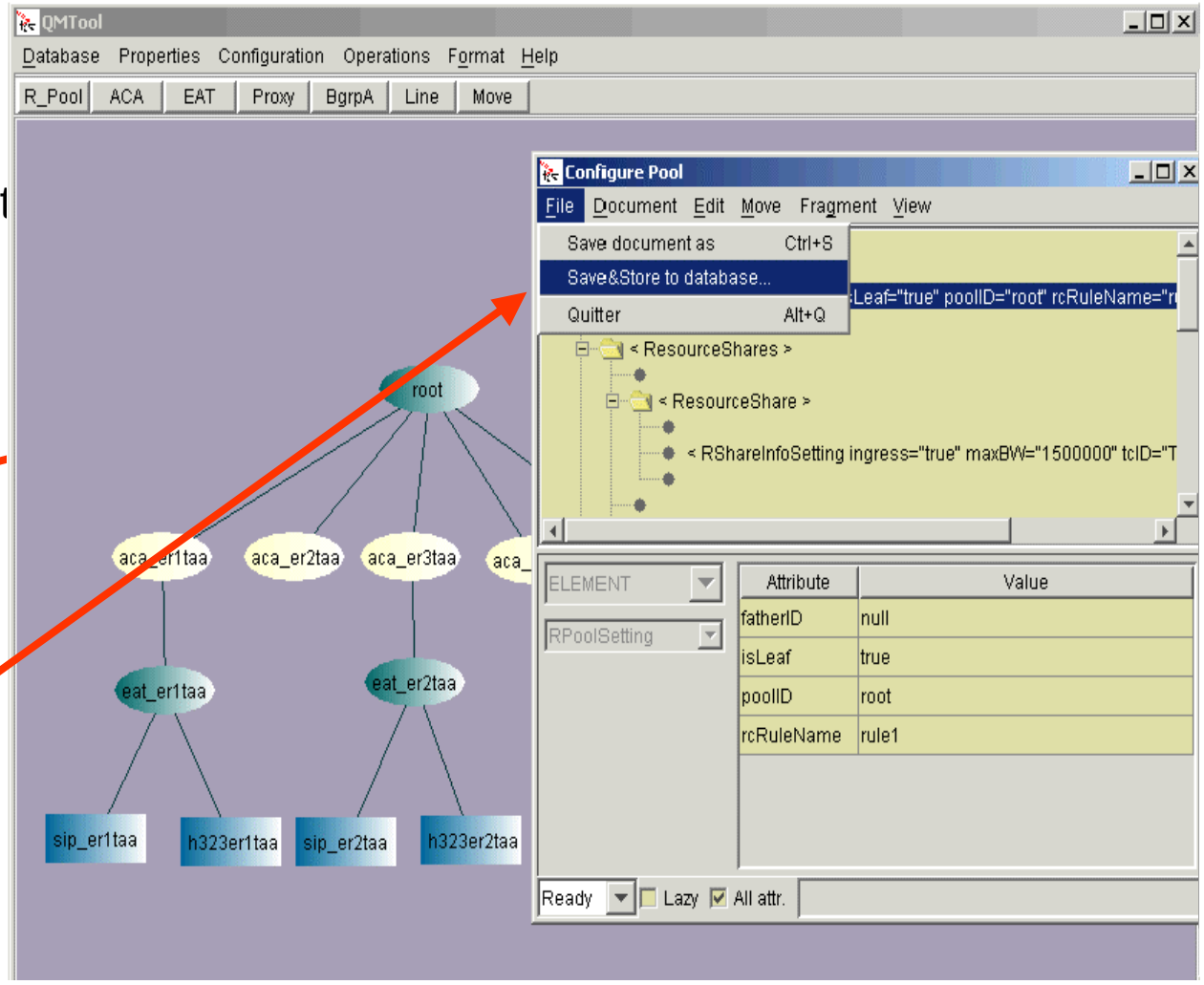
  - The configuration of the components is performed with the use of the XMLOperator's graphical user interface
  - XML files are created and comprise the configuration information of each component – the XML files are based on the DTDs of each component
  - QMTool stores these XML files into the database

  and if I configure faulty a component?

  - QMTool will not let you!
  - QMTool checks the validity of the configuration information in order to be consistent with the design of the network

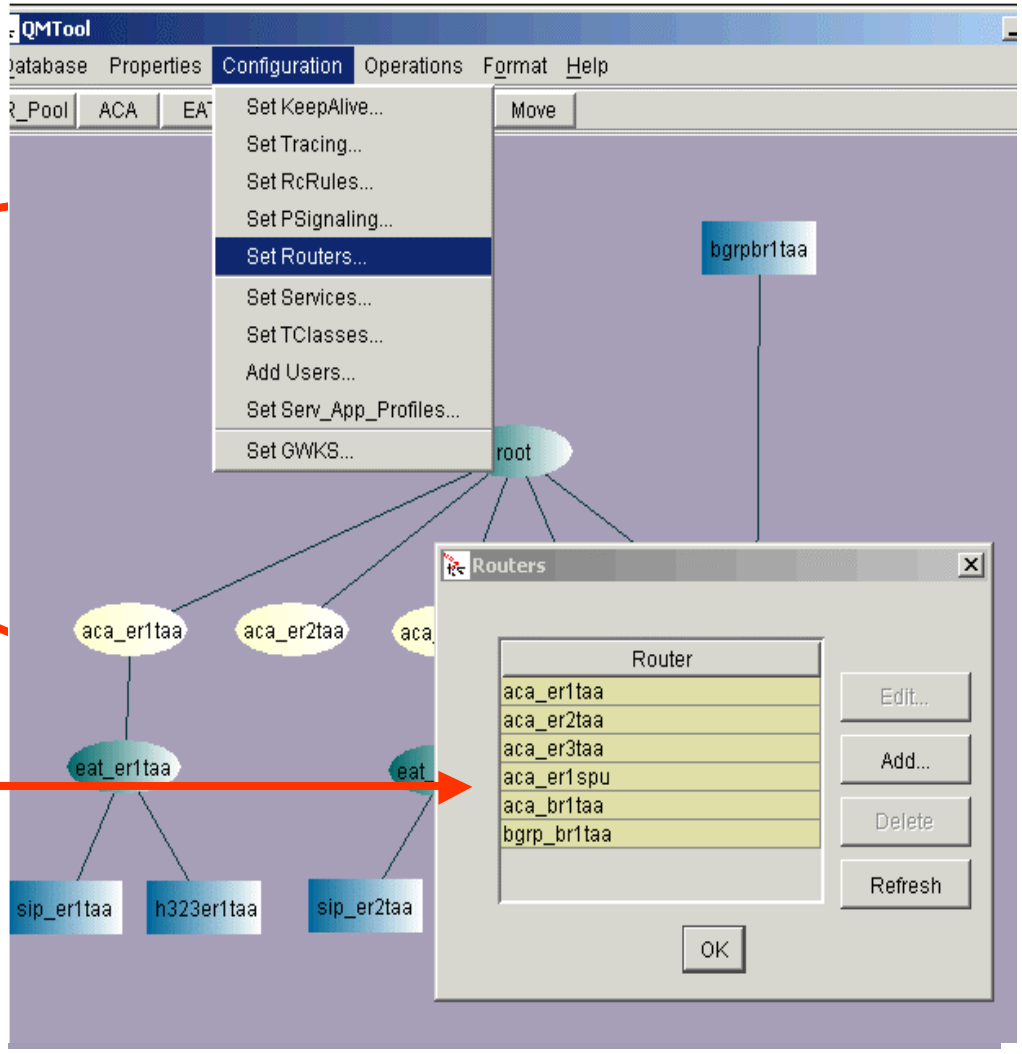
Configuration Demonstration (1)

- Right-click on a component for configuring it or for viewing its configuration
- XMLOperator is used for the editing of the configuration information
- Storage of the configuration information into the database



Configuration Demonstration (2)

- Configuration of non-visualised components, for example KeepAlive mechanism, Network Services, Traffic Classes, Application Profiles, Subscribers – XMLOperator is used
- Dialog Box for the addition, editing, deletion of subscribers
- Likewise, Dialog Box for the addition, editing and deletion of routers



Outline

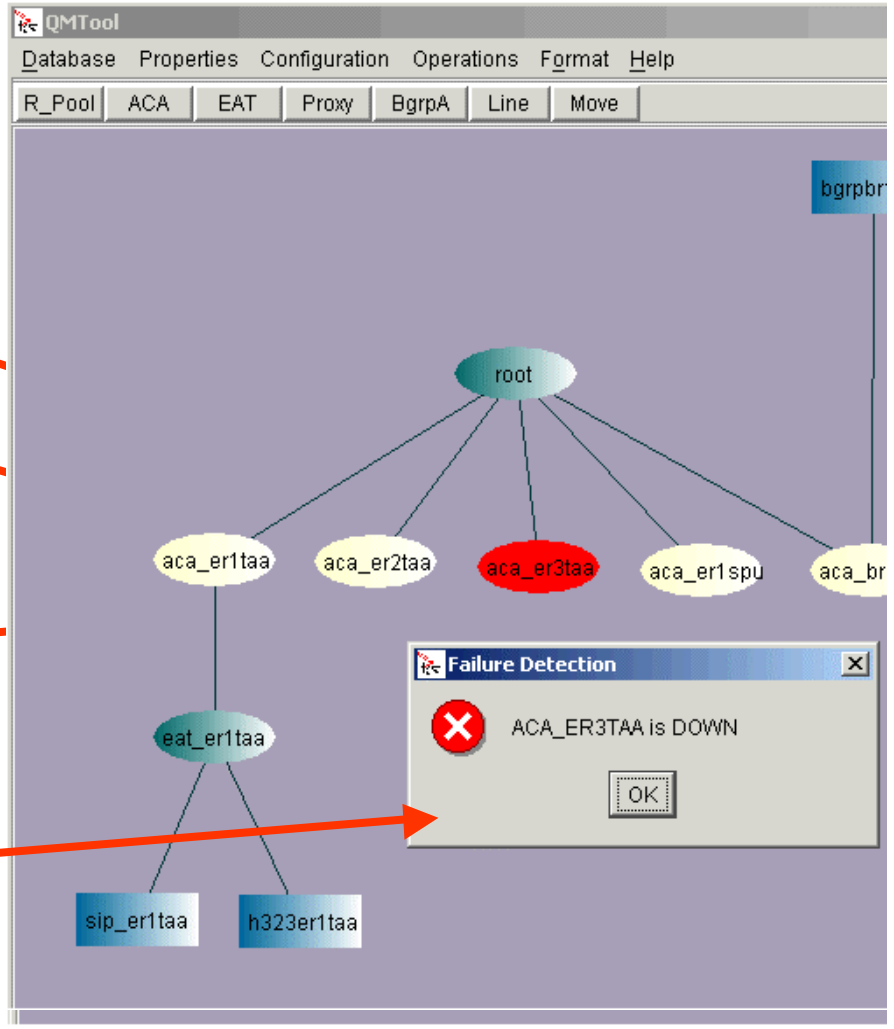
- QMTool approach
- QMTool Functionality
  - Design
  - Configuration
  - Failure Detection
  - Monitoring

Failure Detection

- **QMTool will perform the Failure Detection of the Network Components**
  - The components that their status will be detected (Keep Alive mechanism) are the Resource Pools, ACAs, EATs (RCL) and BGRP Agents (IDOM)
  - The failure detection of a component will have as a result the activation of an alarm mechanism for the notification of the administrator
  - Alarm mechanism – a sound is being played, the color of the component is changed to RED on the QMTool working area

Failure Detection Demonstration

- Right-click on a component for viewing its status
- enable the Failure Detection using the Keep Alive mechanism
- Failure Detection results in changing the component's color and …
- … giving a notification to the administrator



Outline

- QMTool approach
- QMTool Functionality
  - Design
  - Configuration
  - Failure Detection
  - Monitoring

Monitoring

- **QMTool will perform the Monitoring of the Resource Utilisation**

which resources will be monitored?

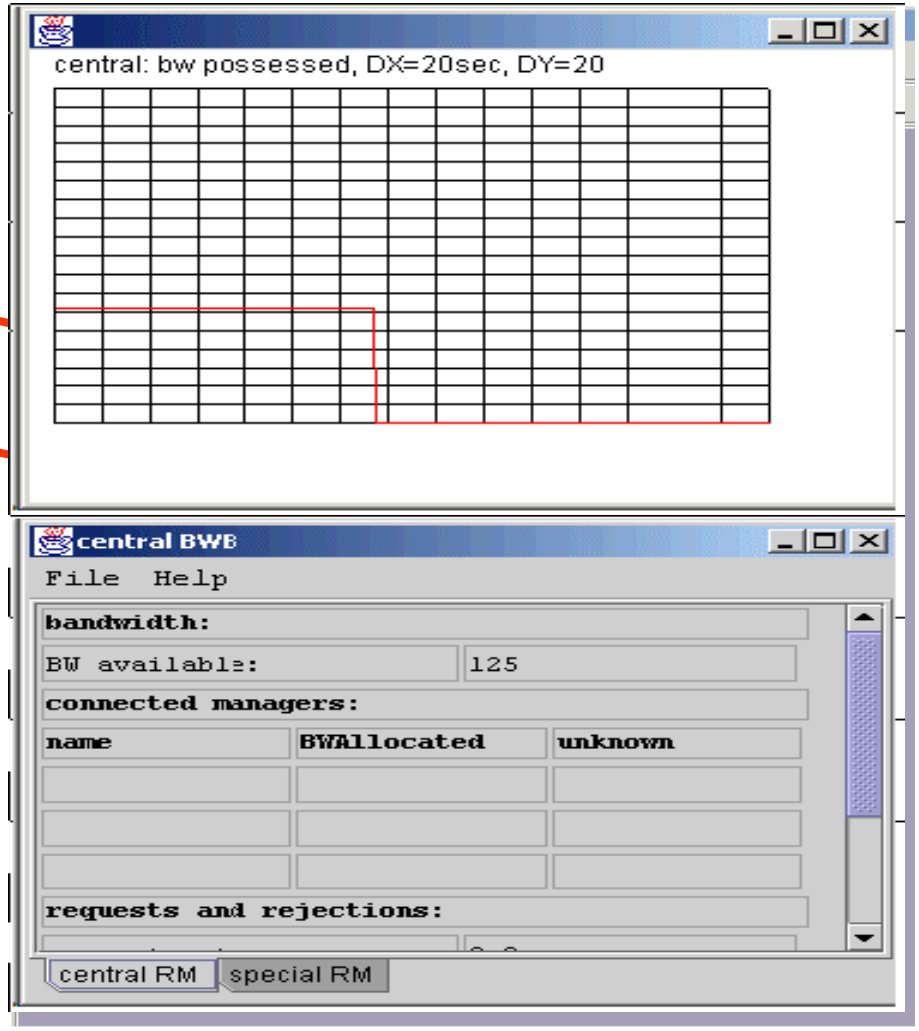
- Two components require monitoring, i.e. the BGRP Agents and the Resource Pools

which is the time interval for the monitoring of the resource utilisation?

- The time interval will be set by the administrator
- The type of monitoring will be continuous
- The administrator will trigger the start of monitoring for a component and the stop of this functionality

Monitoring Demonstration

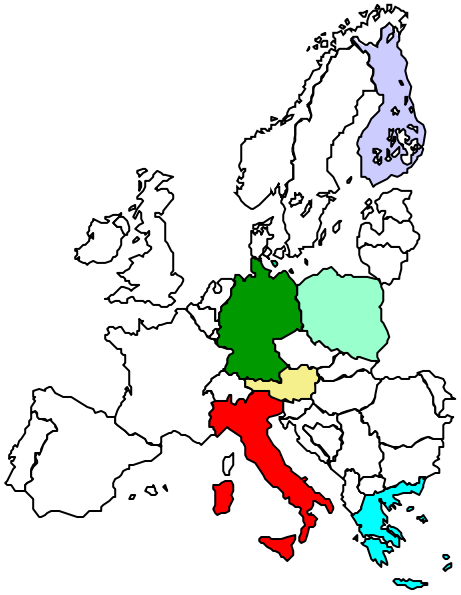
- Right-click on a component (Resource Pool or BGRP Agent) to monitor it
- trigger the monitoring event
- sub-panels will display the requested information in graphical and text



AQUILA (IST-1999-10077)

Adaptive Resource Control for QoS
Using an IP-based Layered Architecture

Project Review No. 4
Premium IP Cluster Workshop



Maastricht, The Netherlands
May 15, 2002

Thank you for your attention !

http://www.ist-aquila.org/