

Global Scheduling and Binding in a Real-Time Embedded Distributed System

**Lisa DiPippo
Victor Fay-Wolfe
Oleg Uvarov
Angela Uvarova**
University of Rhode Island

Trudy Morgan
U.S. Navy SPAWAR Systems Center

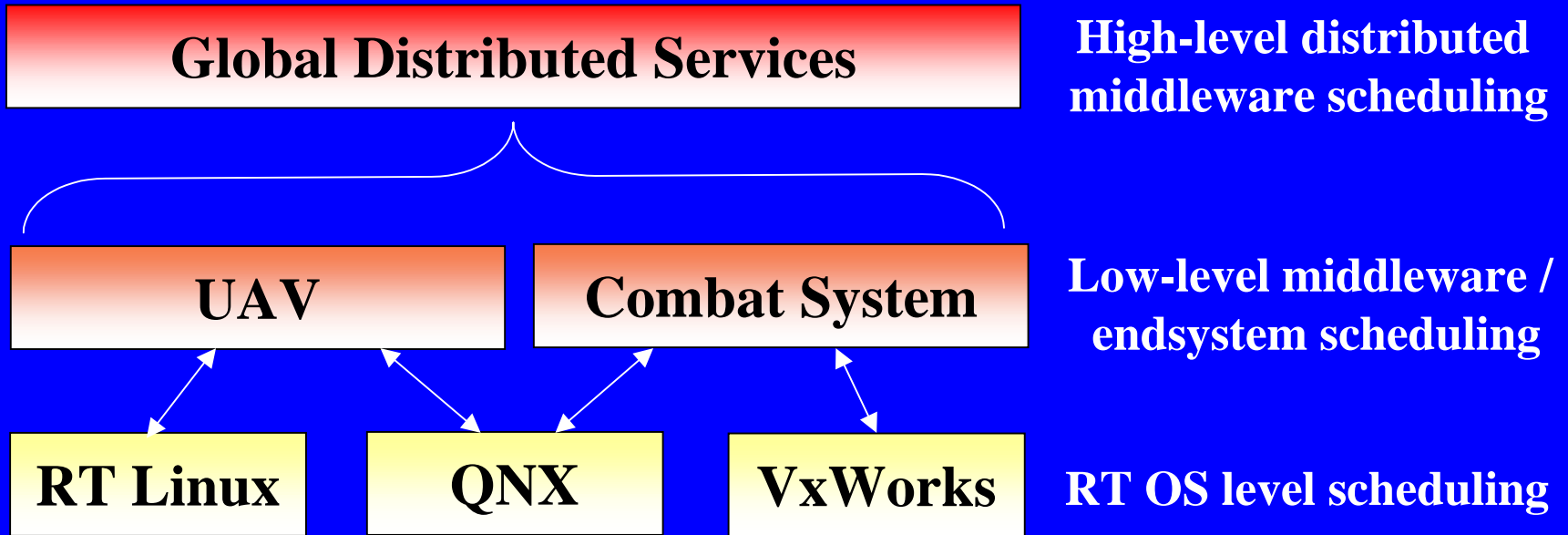
Louis DiPalma
Raytheon

Peter Kortmann
TriPacific Software, Inc.

Presentation Outline

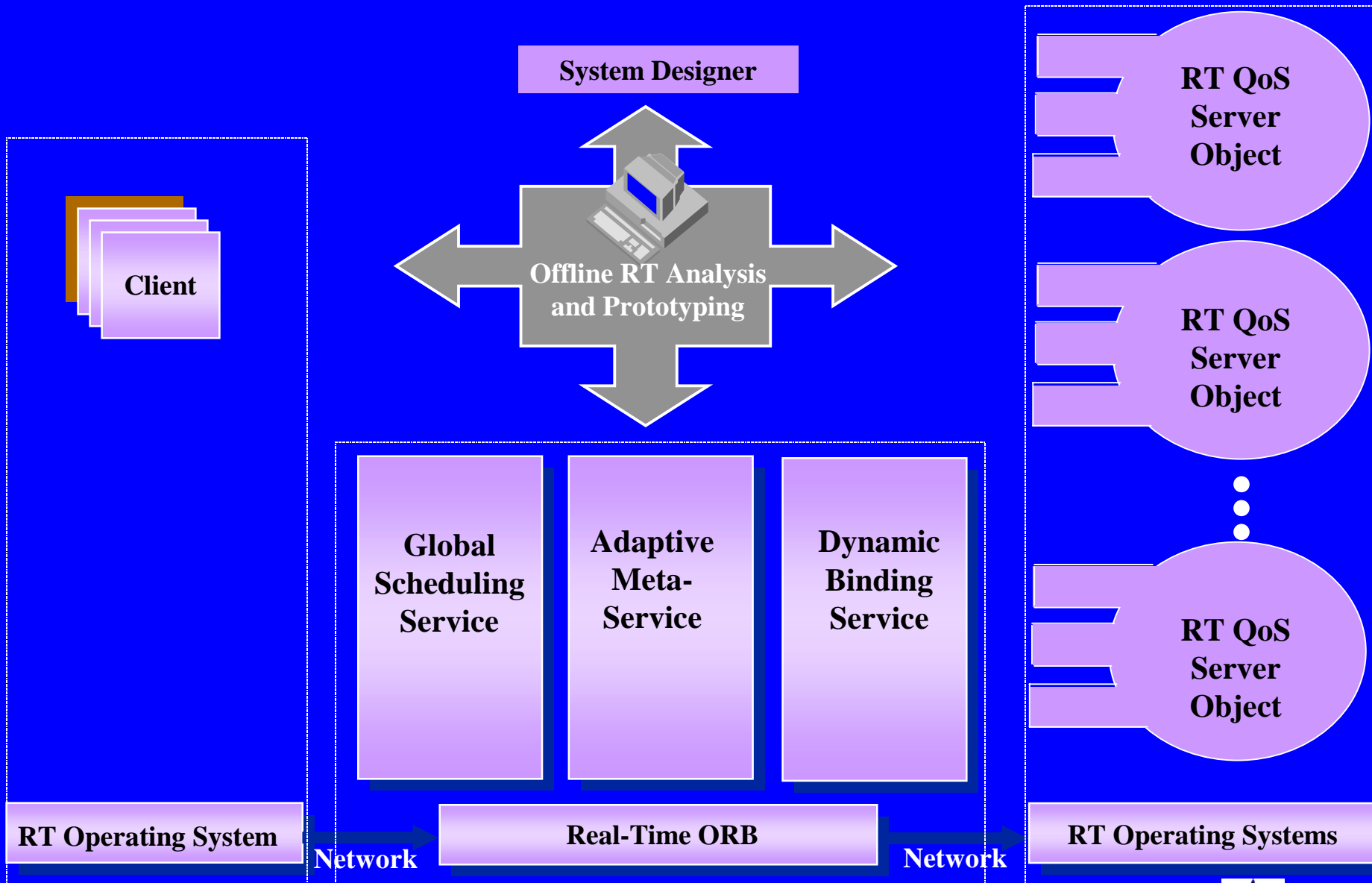
- ◆ Dynamic QoS Services
- ◆ Dynamic QoS Services Architecture
- ◆ Global Scheduling
- ◆ Dynamic Binding
- ◆ Adaptive Meta-Service
- ◆ Application

Dynamic QoS Services

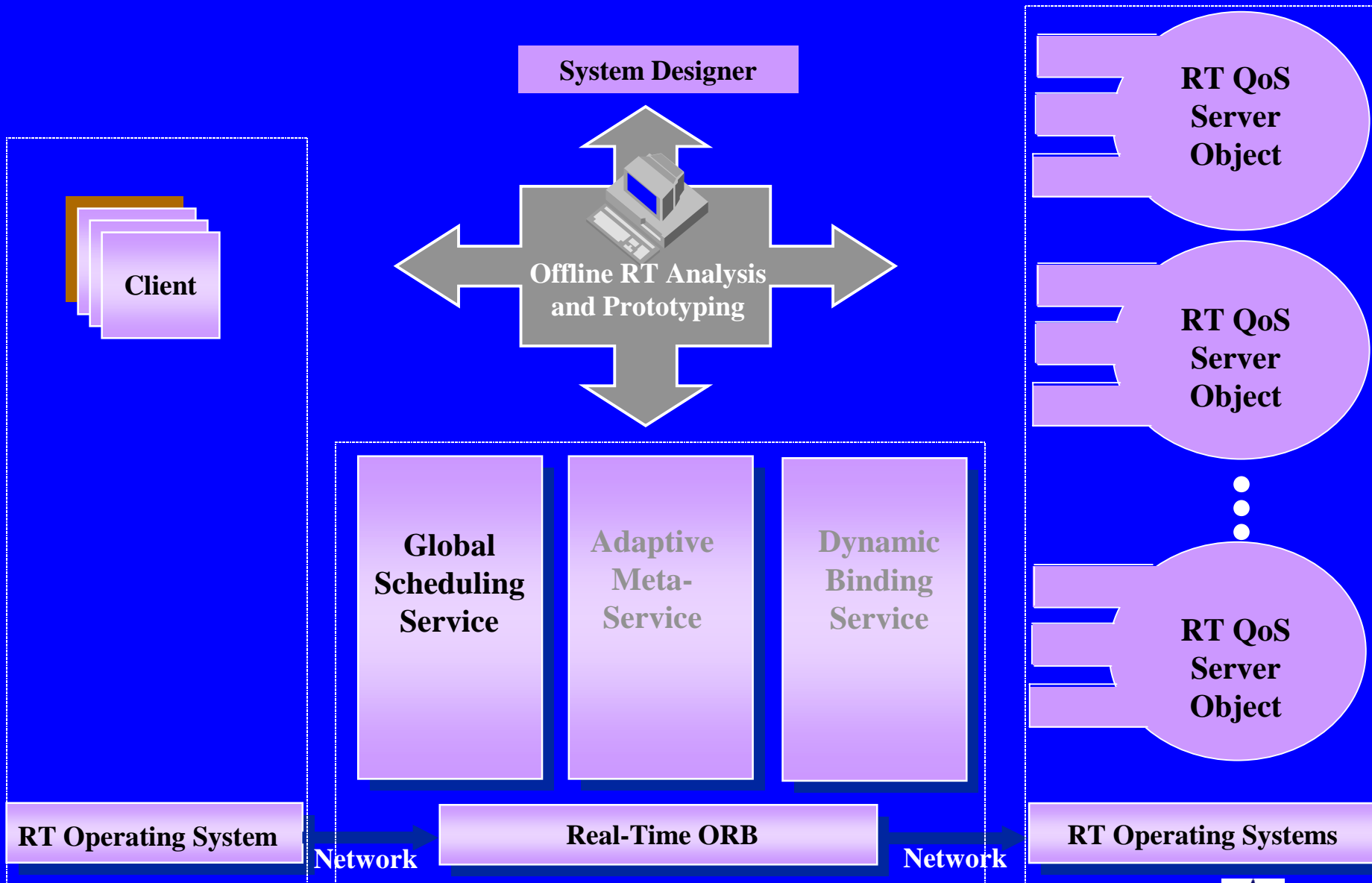


- Work within open systems / COTS
- QoS with focus on real-time

Dynamic QoS Services Architecture



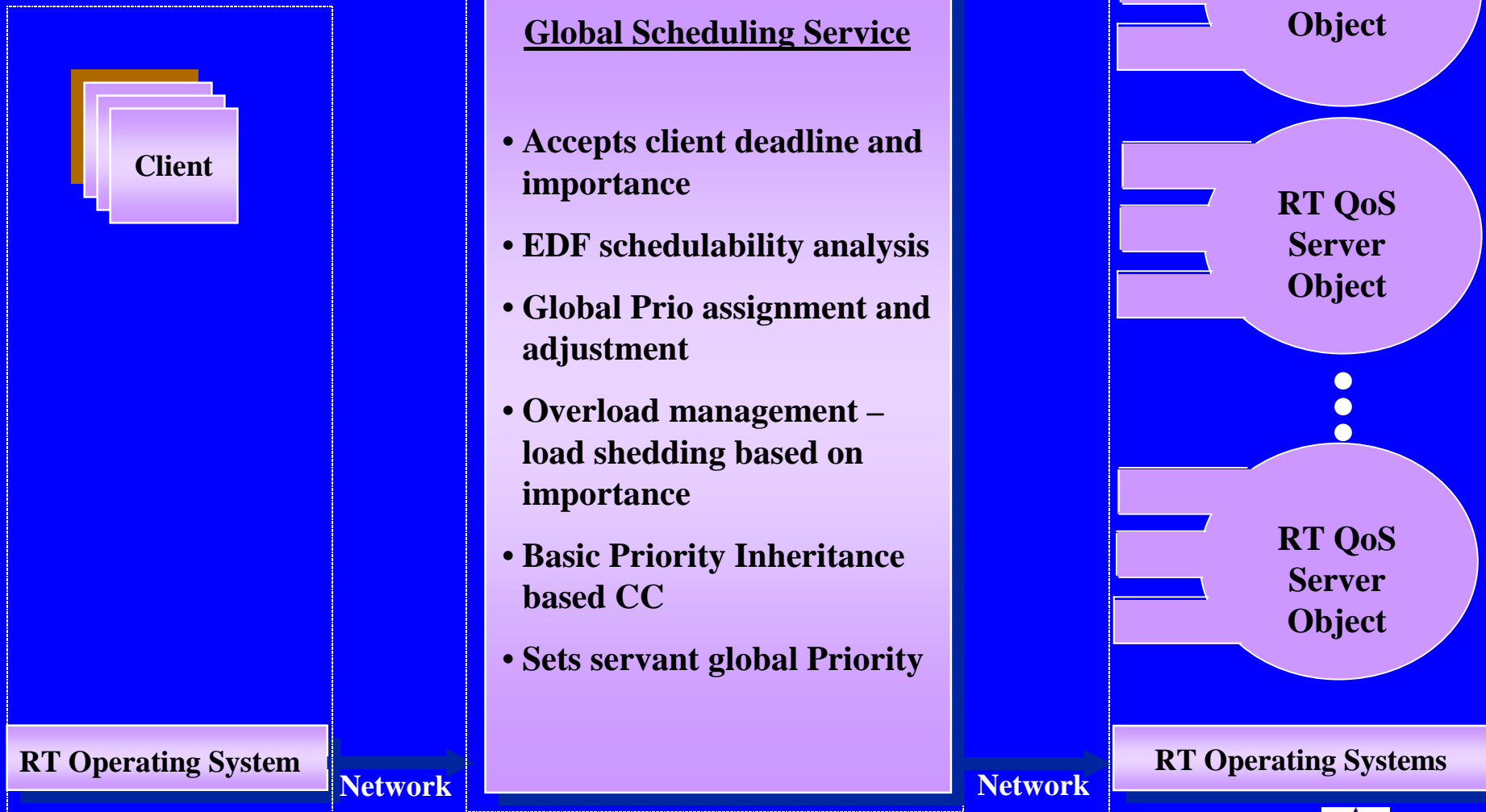
Dynamic QoS Services Architecture



Global Scheduling Service



Global Scheduling Service



Global Scheduling Algorithms

◆ Overload management

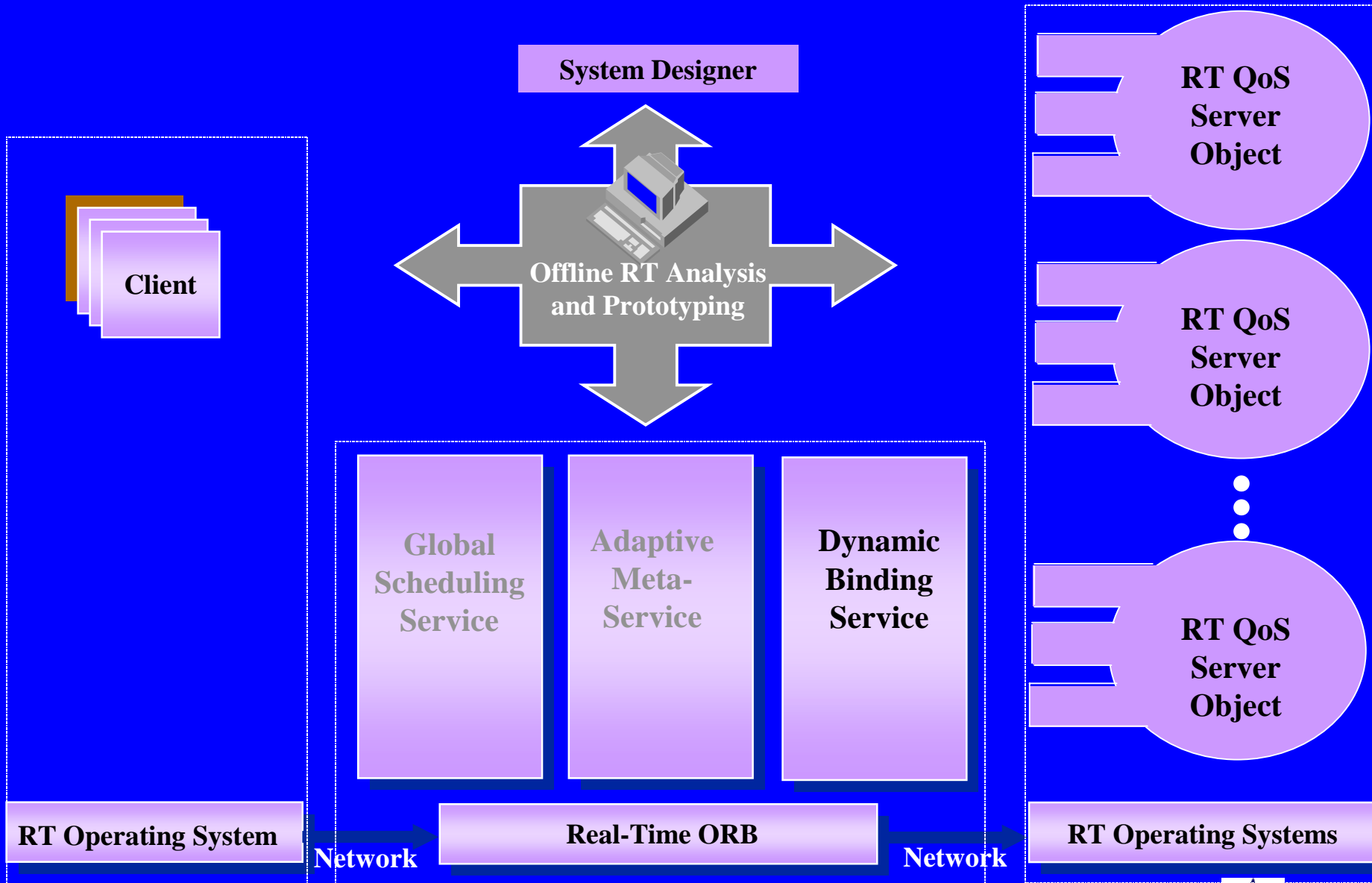
– Load shedding

- » When requested task cannot be scheduled, scheduling service must “shed” one or more tasks
- » Compute *Shedding Parameter* (SP) based upon:
 - ◆ Importance
 - ◆ Remaining execution time
- » Shed task with smallest SP

– Load reduction

- » variation on load shedding
- » reduce execution time and quality of task result instead of shedding

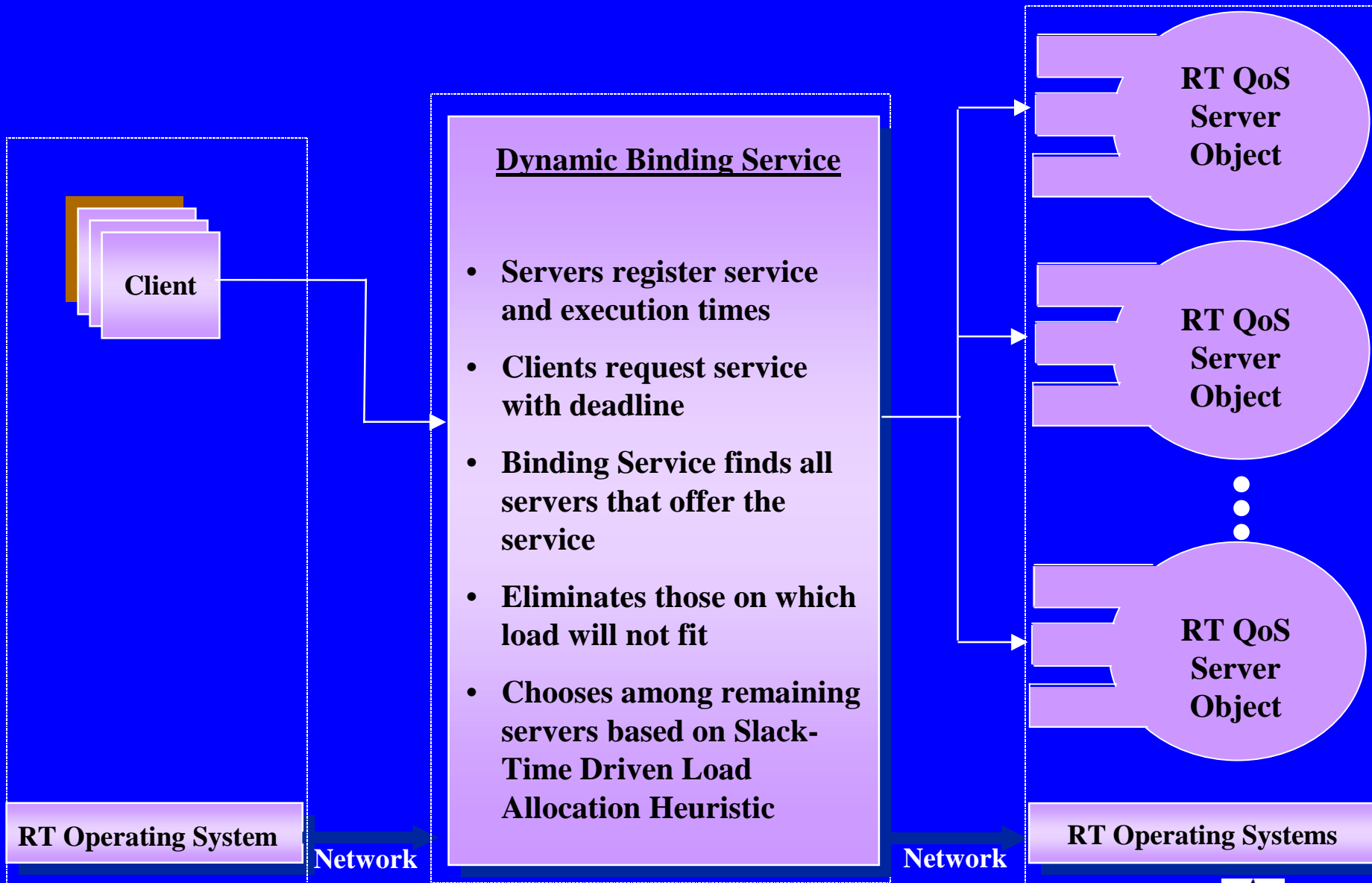
Dynamic QoS Services Architecture



Dynamic Binding Service



Dynamic Binding Service

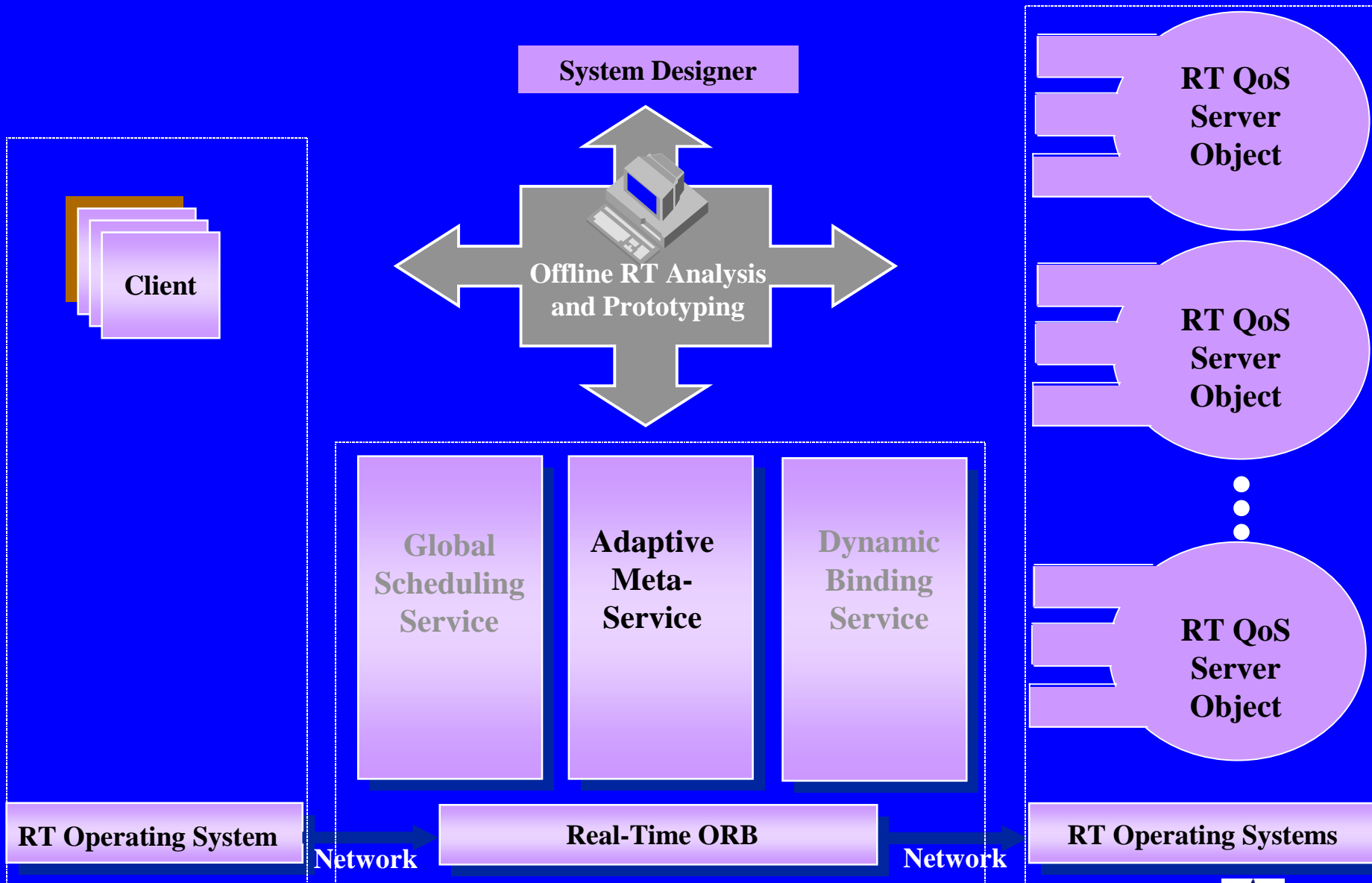


Dynamic Binding Algorithms

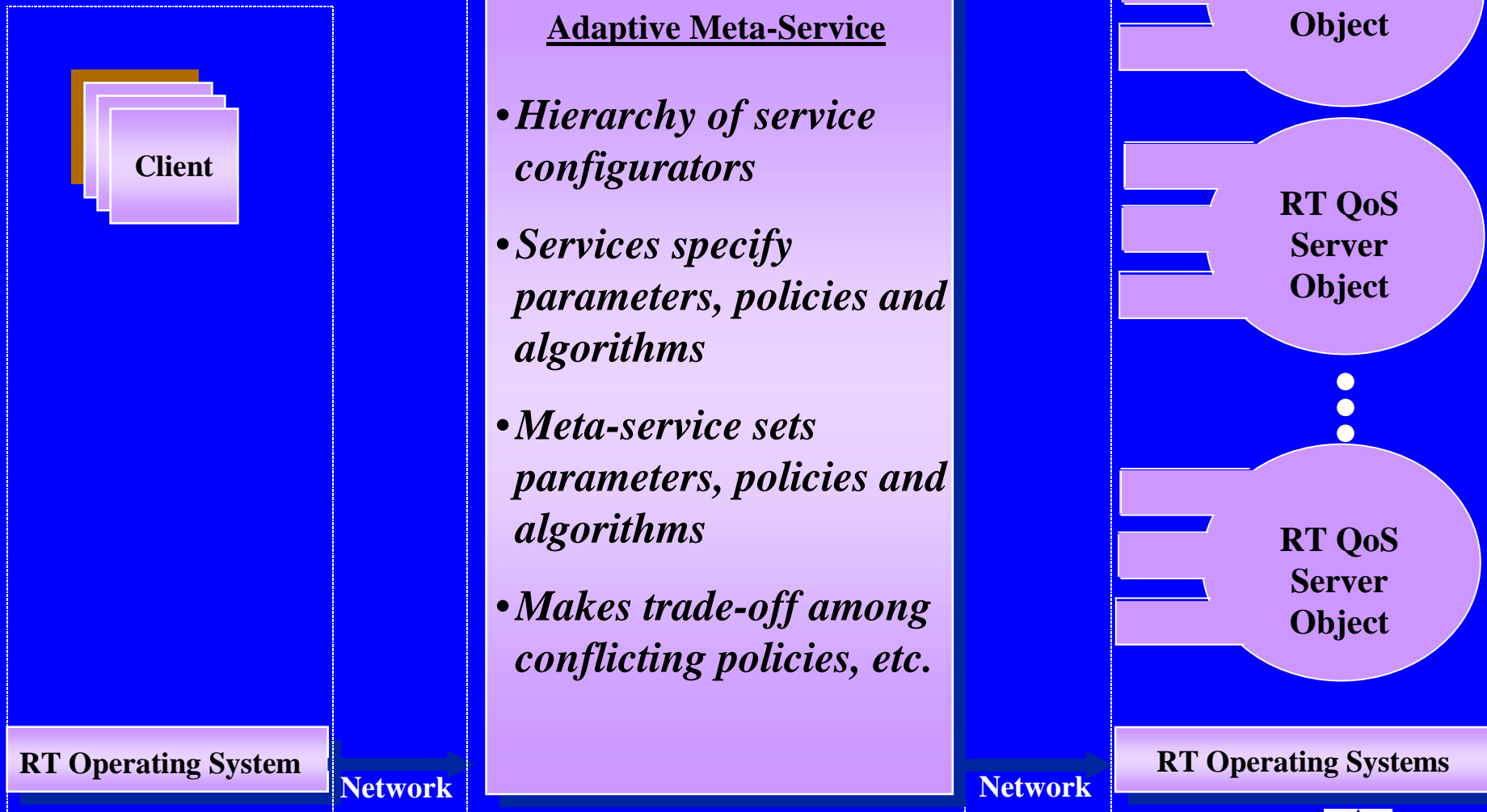
◆ Load allocation

- Goal: Choose the node on which new task fits most tightly, but where tight tasks are finishing soon
- Similar to “Best Fit”
- Computer *Allocation Parameter* (AP) for each candidate node based on:
 - » minimum slack time of all tasks with shorter deadline than new task
 - » deadline of task with minimum slack time
- Choose node with minimum AP

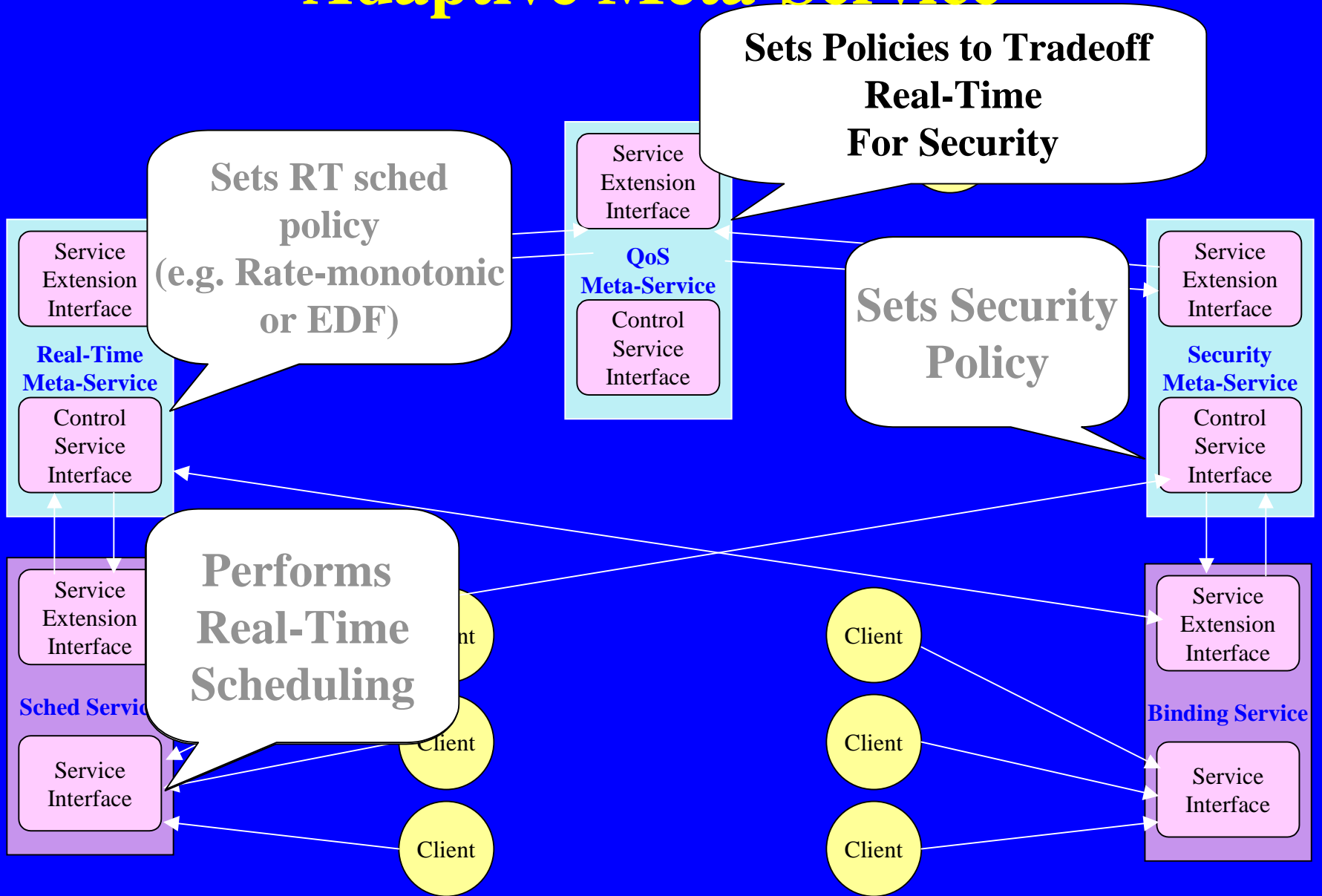
Dynamic QoS Services Architecture



Adaptive Meta-Service



Adaptive Meta-Service



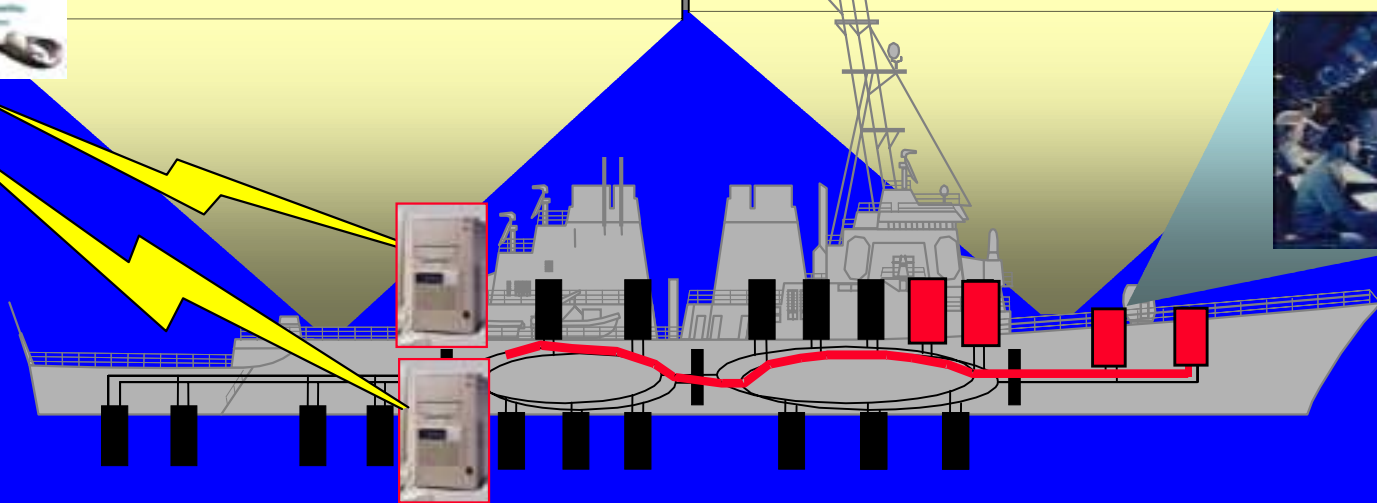
BBN's UAV Application

Dynamic binding

- bind viewer to distributor that will provide best service to viewer

Global Scheduling

- required across distributed system
- schedule tasks and streams
- load shedding to avoid overload



Images courtesy of BBN

Future Directions

- ◆ End-to-end scheduling
 - Scheduling nested requests
 - Load shedding with dependencies
- ◆ Algorithm development
 - Dynamic priority adjustment
 - Dynamic priority mapping
- ◆ Meta-service refinement
 - Continue work on architecture
 - Algorithms for QoS trade-off

Contact Information



homepage.cs.uri.edu/research/rtsorac/

Lisa DiPippo - dipippo@cs.uri.edu

Vic Fay-Wolfe - wolfe@cs.uri.edu