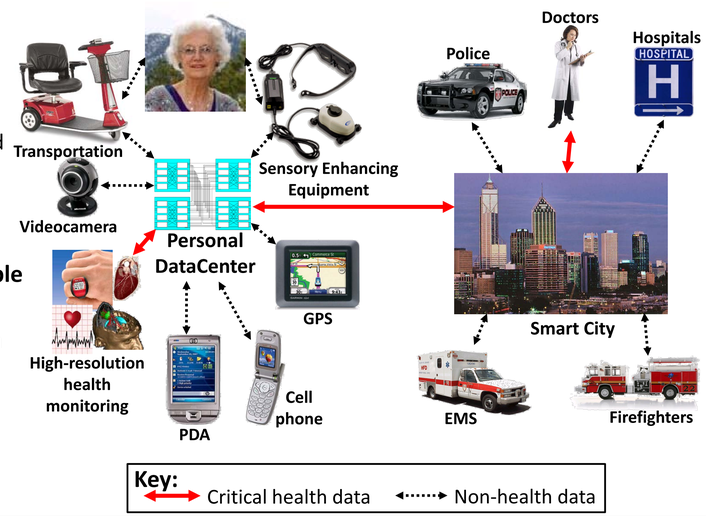
Evaluating and Adapting QoS for Distributed Real-time & Embedded Systems in Dynamic Environments

Joe Hoffert, Vanderbilt University, Nashville, TN, USA

Motivating Example: Smart City Ambient Assisted Living (SCAAL)

- **Scenario**
  - –Smart cities dissolve computational infrastructure, create ubiquitous context-aware services in metropolitan area
  - –Ambient assisted living aids in prolonging and enhancing independent living for the elderly
  - –SCAAL applications empower independent navigation of senior citizens in large cities
- **Personal data center (PDC) manages multiple sensor & data streams**
  - –3-dimensional high-resolution health data
  - –Priorities, update rates change based on patient's health
- **Dynamic environment**



SCAAL Applications Require QoS; Inherently Involve Very Dynamic Environments

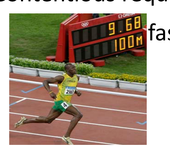
SCAAL Challenges

Timely adaptation to dynamic environments

- Best utilization of resources
- Manual reaction too slow

Managing interacting QoS

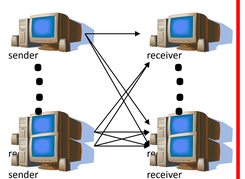
- Contentious requirements



fast & reliable

Scalability

- Number of receivers/senders
- Amount of data



Standardized & robust QoS

- Portability, ease of development
- High configurability
- Tunable

QoS-enabled Pub/Sub Systems in Dynamic Environments Are Challenging to Manage

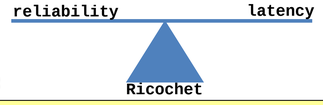
Solution Approach: ADAptive Middleware & Network Transports (ADAMANT)

Timely adaptation to dynamic environments

- Supervised machine learning
- Accurately handle known environments
- Support unknown environments

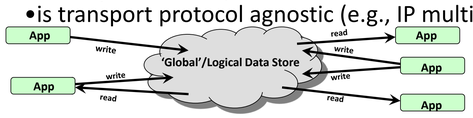
Managing interacting QoS

- Composite metrics evaluate multiple QoS concerns
- Adaptive Network Transport (ANT) framework has composable modules
- Protocols balance QoS (e.g., reliability & latency)



Scalability

- Data Distribution Service:
  - decouples senders, receivers
  - is transport protocol agnostic (e.g., IP multicast)



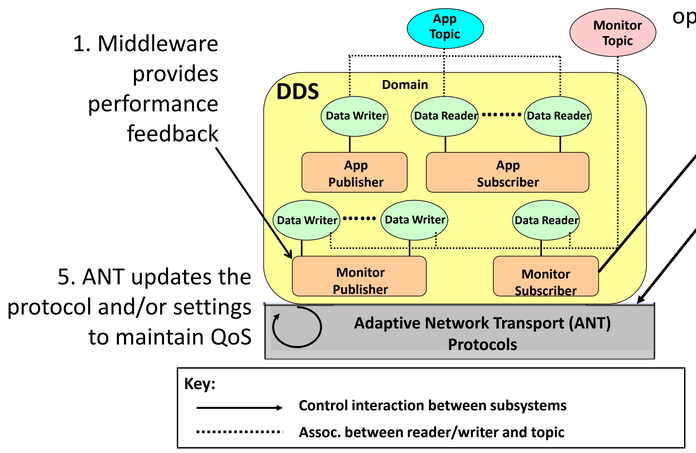
Incorporation of standards

- –Data Distribution Service (DDS) = OMG pub/sub standard
- 22 QoS policies
- Platform-agnostic IDL

ADAMANT addresses the challenges of bounded, timely, scalable adaptation to manage QoS

ADAMANT Architecture & Control Flow

1. Middleware provides performance feedback
2. Controller monitors feedback, sends to optimizer
3. Optimizer responds with optimal protocol & settings
   - –Overfitted ANN provides accurate, constant, & fast response for known envs.
   - –Generalized ANN or SVM provides constant, fast guidance for unknown envs.
4. Controller compares current & optimal settings, notifies ANT as needed
5. ANT updates the protocol and/or settings to maintain QoS



ADAMANT reflects on system state, calculates appropriate changes, & manages adaptation

Future Work

Identify QoS-enabled pub/sub variability

- Reliability; deadlines; data type, amount ; update frequency, data distribution

Develop taxonomies based on variability

Apply DSML to leverage application taxonomies

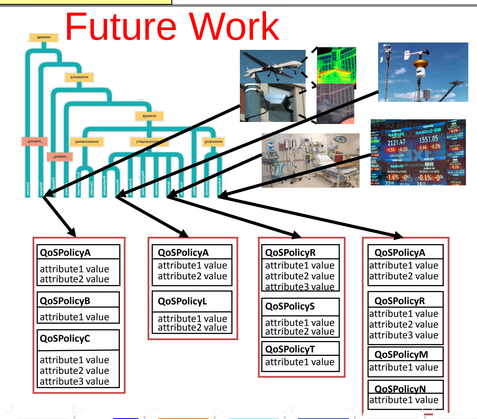
- Map to DDS QoS policies
- Use/create DDS policy patterns

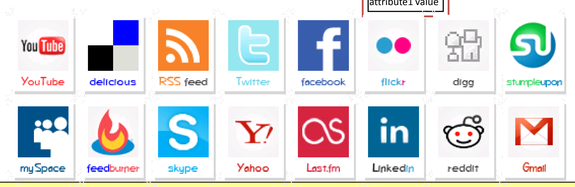
Leverage ADAMANT and Camel DDS

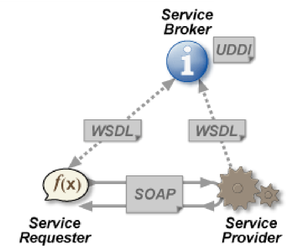
- Camel DDS Component implemented
- Expose QoS interface

Prioritize QoS concerns within composite metrics

- Balance reliability and latency
- Prioritize reliability if possible







mule

Apache Camel

DDS

Further Research:

1. **Specifying contentious requirements via DSML profiles for application types, DDS patterns**
2. **Supporting EJBs, Web Services, ESBs**
3. **Prioritizing QoS aspects within composite metrics**