

Announcement/Subscription/Publication

Message Based Communication for Heterogeneous Mobile Environments

Henry Ristau

University of Rostock
Information and Communication Services Group
Rostock, Germany

29.04.2009



Outline

Introduction

Announcement/Subscription/Publication

System Architecture

Routing algorithm

Evaluation

Conclusion and Future Work



Outline

Introduction

Announcement/Subscription/Publication

System Architecture

Routing algorithm

Evaluation

Conclusion and Future Work



Heterogeneity

What does heterogeneity mean for smart environments?

- ▶ Incompatible communication techniques
 - ▶ No homogeneous addressing scheme
 - ▶ No guarantee for unique addresses
- ▶ Heterogeneous devices
- ▶ Heterogeneous applications
 - ▶ Different document / data formats
 - ▶ Different context



Heterogeneity

What does heterogeneity mean for smart environments?

- ▶ Incompatible communication techniques
 - ▶ No homogeneous addressing scheme
 - ▶ No guarantee for unique addresses
- ▶ Heterogeneous devices
- ▶ Heterogeneous applications
 - ▶ Different document / data formats
 - ▶ Different context



Heterogeneity

What does heterogeneity mean for smart environments?

- ▶ Incompatible communication techniques
 - ▶ No homogeneous addressing scheme
 - ▶ No guarantee for unique addresses
- ▶ Heterogeneous devices
- ▶ Heterogeneous applications
 - ▶ Different document / data formats
 - ▶ Different context



Heterogeneity

What does heterogeneity mean for smart environments?

- ▶ Incompatible communication techniques
 - ▶ No homogeneous addressing scheme
 - ▶ No guarantee for unique addresses
- ▶ Heterogeneous devices
- ▶ Heterogeneous applications
 - ▶ Different document / data formats
 - ▶ Different context



Requirements

What requirements do smart environments demand from networking?

- ▶ Transparency from network topology
- ▶ Availability of information
- ▶ Independency of applications to the network / middleware
- ▶ Transparent aggregation / conversion / processing of data
- ▶ Decoupling in **space**
- ▶ Decoupling in **time**
- ▶ Decoupling in **threads**
- ▶ Decoupling in **semantics**



Requirements

What requirements do smart environments demand from networking?

- ▶ Transparency from network topology
- ▶ Availability of information
- ▶ Independency of applications to the network / middleware
- ▶ Transparent aggregation / conversion / processing of data
- ▶ Decoupling in **space**
- ▶ Decoupling in **time**
- ▶ Decoupling in **threads**
- ▶ Decoupling in **semantics**



Requirements

What requirements do smart environments demand from networking?

- ▶ Transparency from network topology
- ▶ Availability of information
- ▶ Independency of applications to the network / middleware
- ▶ Transparent aggregation / conversion / processing of data
- ▶ Decoupling in **space**
- ▶ Decoupling in **time**
- ▶ Decoupling in **threads**
- ▶ Decoupling in **semantics**



Requirements

What requirements do smart environments demand from networking?

- ▶ Transparency from network topology
- ▶ Availability of information
- ▶ Independency of applications to the network / middleware
- ▶ Transparent aggregation / conversion / processing of data
- ▶ Decoupling in **space**
- ▶ Decoupling in **time**
- ▶ Decoupling in **threads**
- ▶ Decoupling in **semantics**



Requirements

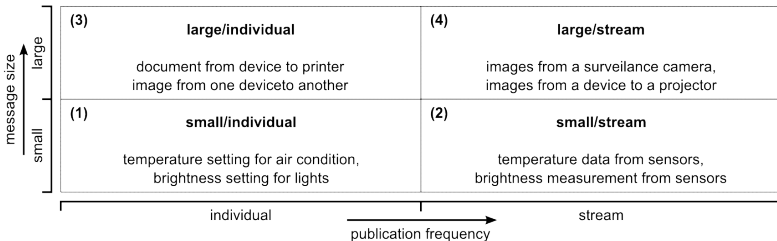
What requirements do smart environments demand from networking?

- ▶ Transparency from network topology
- ▶ Availability of information
- ▶ Independency of applications to the network / middleware
- ▶ Transparent aggregation / conversion / processing of data
- ▶ Decoupling in **space**
- ▶ Decoupling in **time**
- ▶ Decoupling in **threads**
- ▶ Decoupling in **semantics**



Applications Scenarios

- ▶ Classification by message size and publication frequency
- ▶ focus on class 2 - small/stream



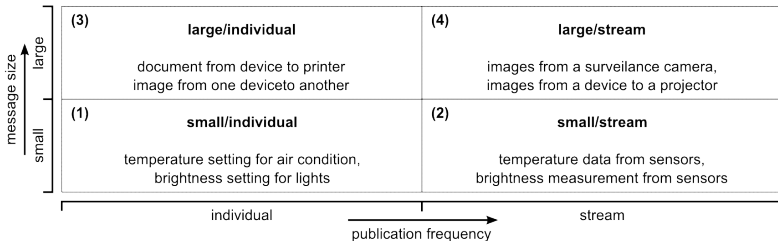
- ▶ small \Rightarrow size \leq MTU
- ▶ large \Rightarrow size $>$ MTU

- ▶ individual \Rightarrow single message
- ▶ stream \Rightarrow multiple messages



Applications Scenarios

- ▶ Classification by message size and publication frequency
- ▶ focus on class 2 - small/stream



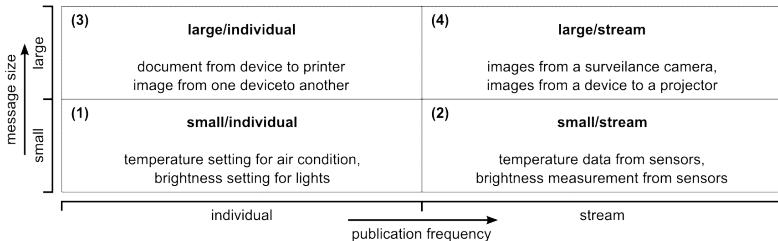
- ▶ small \Rightarrow size \leq MTU
- ▶ large \Rightarrow size $>$ MTU

- ▶ individual \Rightarrow single message
- ▶ stream \Rightarrow multiple messages



Applications Scenarios

- ▶ Classification by message size and publication frequency
- ▶ focus on class 2 - small/stream



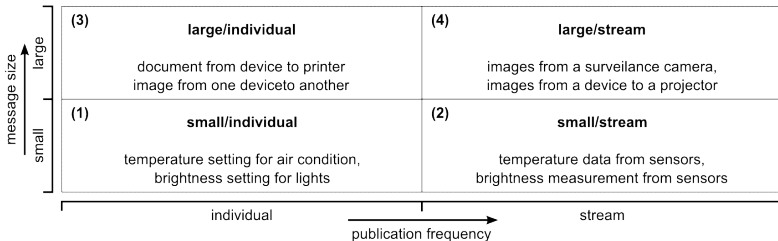
- ▶ small \Rightarrow size \leq MTU
- ▶ large \Rightarrow size $>$ MTU

- ▶ individual \Rightarrow single message
- ▶ stream \Rightarrow multiple messages



Applications Scenarios

- ▶ Classification by message size and publication frequency
- ▶ focus on class 2 - small/stream



▶ small \Rightarrow size \leq MTU

▶ large \Rightarrow size $>$ MTU

▶ individual \Rightarrow single message

▶ stream \Rightarrow multiple messages



Outline

Introduction

Announcement/Subscription/Publication

System Architecture

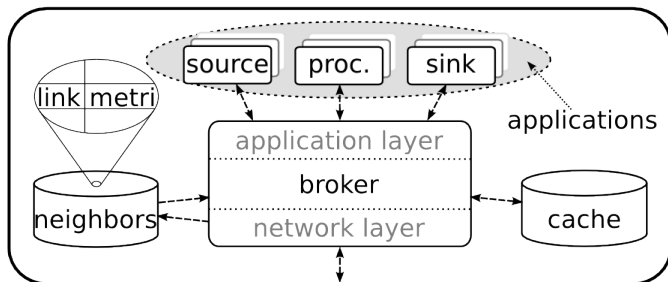
Routing algorithm

Evaluation

Conclusion and Future Work



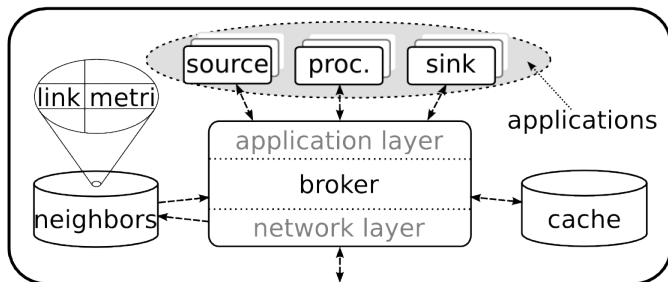
Basic System Architecture



- ▶ 3 types of applications: sources, processors, sinks
- ▶ One broker per device
 - ▶ Decouples applications from each other and the network
 - ▶ Network communication to "neighboring" brokers



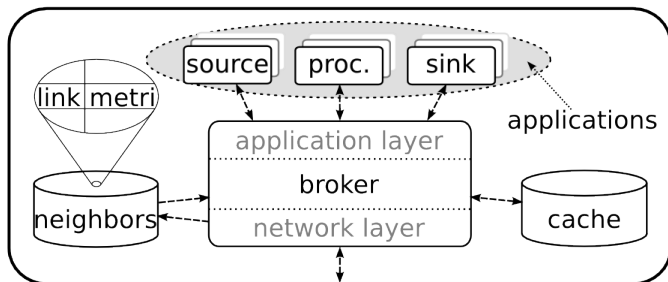
Basic System Architecture



- ▶ 3 types of applications: sources, processors, sinks
- ▶ One broker per device
 - ▶ Decouples applications from each other and the network
 - ▶ Network communication to "neighboring" brokers



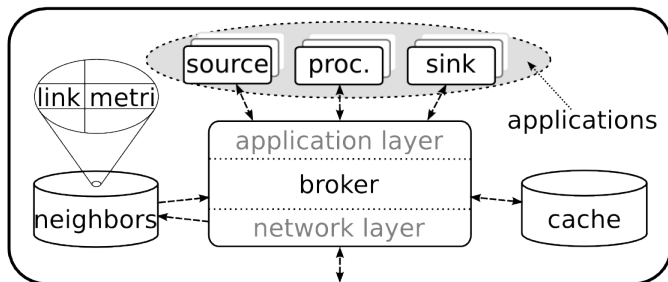
Basic System Architecture



- ▶ 3 types of applications: sources, processors, sinks
- ▶ One broker per device
 - ▶ Decouples applications from each other and the network
 - ▶ Network communication to “neighboring” brokers



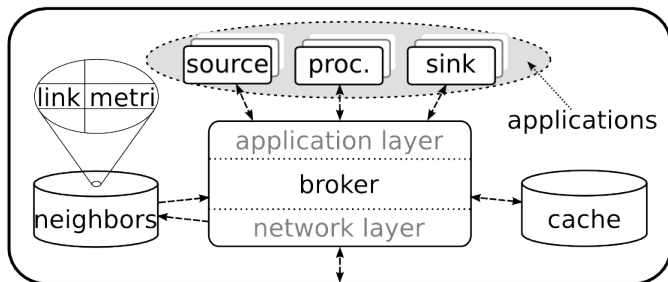
Basic System Architecture



- ▶ 3 types of applications: sources, processors, sinks
- ▶ One broker per device
 - ▶ Decouples applications from each other and the network
 - ▶ Network communication to “neighboring” brokers



Basic System Architecture



- ▶ 3 types of applications: sources, processors, sinks
- ▶ One broker per device
 - ▶ Decouples applications from each other and the network
 - ▶ Network communication to “neighboring” brokers



ASP - Layers

▶ Applications

- ▶ Source
- ▶ Sink
- ▶ Processor

▶ ASP Routing algorithm

▶ Network Abstraction Layer

- ▶ Neighbor discovery
- ▶ Neighbor cache updates
(address, metric)
- ▶ Reliable message delivery
- ▶ MTU definition

▶ Interface

- ▶ register
- ▶ publish
- ▶ receive announcements
- ▶ subscribe
- ▶ send announcements
- ▶ receive publications
- ▶ send publications



ASP - Layers

▶ Applications

- ▶ Source
- ▶ Sink
- ▶ Processor

▶ ASP Routing algorithm

▶ Network Abstraction Layer

- ▶ Neighbor discovery
- ▶ Neighbor cache updates (address, metric)
- ▶ Reliable message delivery
- ▶ MTU definition

▶ Interface

- ▶ **register**
- ▶ **publish**
- ▶ receive announcements
- ▶ subscribe
- ▶ send announcements
- ▶ receive publications
- ▶ send publications



ASP - Layers

▶ Applications

- ▶ Source
- ▶ Sink
- ▶ Processor

▶ ASP Routing algorithm

▶ Network Abstraction Layer

- ▶ Neighbor discovery
- ▶ Neighbor cache updates (address, metric)
- ▶ Reliable message delivery
- ▶ MTU definition

▶ Interface

- ▶ **register**
- ▶ publish
- ▶ **receive announcements**
- ▶ **subscribe**
- ▶ send announcements
- ▶ **receive publications**
- ▶ send publications



ASP - Layers

- ▶ Applications
 - ▶ Source
 - ▶ Sink
 - ▶ Processor
- ▶ ASP Routing algorithm
- ▶ Network Abstraction Layer
 - ▶ Neighbor discovery
 - ▶ Neighbor cache updates (address, metric)
 - ▶ Reliable message delivery
 - ▶ MTU definition
- ▶ Interface
 - ▶ **register**
 - ▶ publish
 - ▶ **receive announcements**
 - ▶ subscribe
 - ▶ **send announcements**
 - ▶ **receive publications**
 - ▶ **send publications**



ASP - Layers

- ▶ Applications
 - ▶ Source
 - ▶ Sink
 - ▶ Processor
- ▶ ASP Routing algorithm
- ▶ Network Abstraction Layer
 - ▶ Neighbor discovery
 - ▶ Neighbor cache updates (address, metric)
 - ▶ Reliable message delivery
 - ▶ MTU definition
- ▶ Interface
 - ▶ register
 - ▶ publish
 - ▶ receive announcements
 - ▶ subscribe
 - ▶ send announcements
 - ▶ receive publications
 - ▶ send publications



ASP - Layers

▶ Applications

- ▶ Source
- ▶ Sink
- ▶ Processor

▶ ASP Routing algorithm

▶ Network Abstraction Layer

- ▶ Neighbor discovery
- ▶ Neighbor cache updates
(address, metric)
- ▶ Reliable message delivery
- ▶ MTU definition

▶ Interface

- ▶ register
- ▶ publish
- ▶ receive announcements
- ▶ subscribe
- ▶ send announcements
- ▶ receive publications
- ▶ send publications



ASP - Layers

- ▶ Applications
 - ▶ Source
 - ▶ Sink
 - ▶ Processor
- ▶ ASP Routing algorithm
- ▶ Network Abstraction Layer
 - ▶ Neighbor discovery
 - ▶ Neighbor cache updates (address, metric)
 - ▶ Reliable message delivery
 - ▶ MTU definition
- ▶ Interface
 - ▶ register
 - ▶ publish
 - ▶ receive announcements
 - ▶ subscribe
 - ▶ send announcements
 - ▶ receive publications
 - ▶ send publications



ASP - Layers

- ▶ Applications
 - ▶ Source
 - ▶ Sink
 - ▶ Processor
- ▶ ASP Routing algorithm
- ▶ Network Abstraction Layer
 - ▶ Neighbor discovery
 - ▶ Neighbor cache updates (address, metric)
 - ▶ Reliable message delivery
 - ▶ MTU definition
- ▶ Interface
 - ▶ register
 - ▶ publish
 - ▶ receive announcements
 - ▶ subscribe
 - ▶ send announcements
 - ▶ receive publications
 - ▶ send publications



ASP - Layers

- ▶ Applications
 - ▶ Source
 - ▶ Sink
 - ▶ Processor
- ▶ ASP Routing algorithm
- ▶ Network Abstraction Layer
 - ▶ Neighbor discovery
 - ▶ Neighbor cache updates (address, metric)
 - ▶ Reliable message delivery
 - ▶ MTU definition
- ▶ Interface
 - ▶ register
 - ▶ publish
 - ▶ receive announcements
 - ▶ subscribe
 - ▶ send announcements
 - ▶ receive publications
 - ▶ send publications

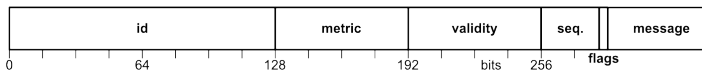


ASP - Layers

- ▶ Applications
 - ▶ Source
 - ▶ Sink
 - ▶ Processor
- ▶ ASP Routing algorithm
- ▶ Network Abstraction Layer
 - ▶ Neighbor discovery
 - ▶ Neighbor cache updates (address, metric)
 - ▶ Reliable message delivery
 - ▶ MTU definition
- ▶ Interface
 - ▶ register
 - ▶ publish
 - ▶ receive announcements
 - ▶ subscribe
 - ▶ send announcements
 - ▶ receive publications
 - ▶ send publications



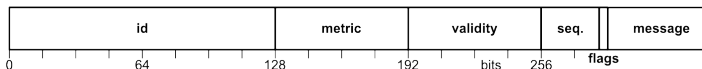
Phase 1: Announcement



- ▶ Task
 - ▶ Propagation of information availability
- ▶ Distribution
 - ▶ Flooding from source
 - ▶ ... through all processors
 - ▶ ... keeping best path metric
- ▶ Content
 - ▶ First dataset or special dataset
 - ▶ Size \leq MTU



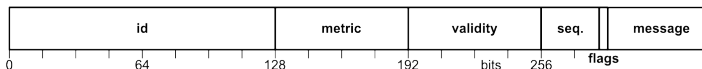
Phase 1: Announcement



- ▶ Task
 - ▶ Propagation of information availability
- ▶ Distribution
 - ▶ Flooding from source
 - ▶ ... through all processors
 - ▶ ... keeping best path metric
- ▶ Content
 - ▶ First dataset or special dataset
 - ▶ Size \leq MTU



Phase 1: Announcement



- ▶ Task
 - ▶ Propagation of information availability
- ▶ Distribution
 - ▶ Flooding from source
 - ▶ ... through all processors
 - ▶ ... keeping best path metric
- ▶ Content
 - ▶ First dataset or special dataset
 - ▶ Size \leq MTU



Phase 2: Subscription



▶ Task

▶ Control messages

- ▶ Subscribe/Unsubscribe from sink towards source
- ▶ Broken path signalling from breakage towards source

▶ Distribution

- ▶ Best path according to path metric
- ▶ Path becomes active path on subscribe

▶ Content

- ▶ Only control message
- ▶ Size \ll MTU



Phase 2: Subscription



- ▶ Task
 - ▶ Control messages
 - ▶ Subscribe/Unsubscribe from sink towards source
 - ▶ Broken path signalling from breakage towards source
- ▶ Distribution
 - ▶ Best path according to path metric
 - ▶ Path becomes active path on subscribe
- ▶ Content
 - ▶ Only control message
 - ▶ Size \ll MTU



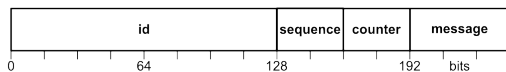
Phase 2: Subscription



- ▶ Task
 - ▶ Control messages
 - ▶ Subscribe/Unsubscribe from sink towards source
 - ▶ Broken path signalling from breakage towards source
- ▶ Distribution
 - ▶ Best path according to path metric
 - ▶ Path becomes active path on subscribe
- ▶ Content
 - ▶ Only control message
 - ▶ Size \ll MTU



Phase 3: Publication



- ▶ Task
 - ▶ Information delivery
- ▶ Distribution
 - ▶ Single publication per active path
- ▶ Content
 - ▶ Dataset
 - ▶ Size \leq MTU (only in class 2)
 - ▶ Otherwise fragmentation



Phase 3: Publication



- ▶ Task
 - ▶ Information delivery
- ▶ Distribution
 - ▶ Single publication per active path
- ▶ Content
 - ▶ Dataset
 - ▶ Size \leq MTU (only in class 2)
 - ▶ Otherwise fragmentation



Phase 3: Publication



- ▶ Task
 - ▶ Information delivery

- ▶ Distribution
 - ▶ Single publication per active path

- ▶ Content
 - ▶ Dataset
 - ▶ Size \leq MTU (only in class 2)
 - ▶ Otherwise fragmentation



Outline

Introduction

Announcement/Subscription/Publication

System Architecture

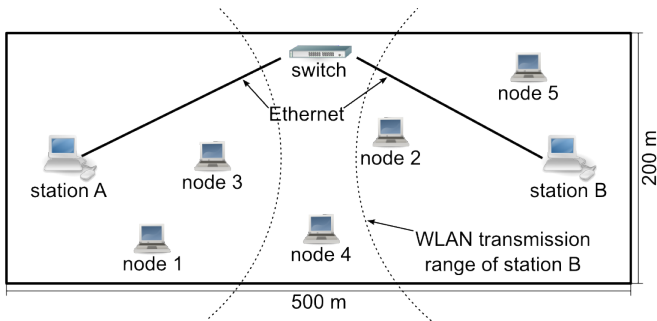
Routing algorithm

Evaluation

Conclusion and Future Work



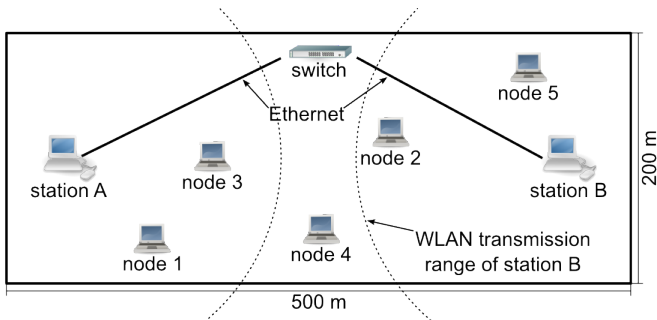
Methodology



► Simple very mobile scenario

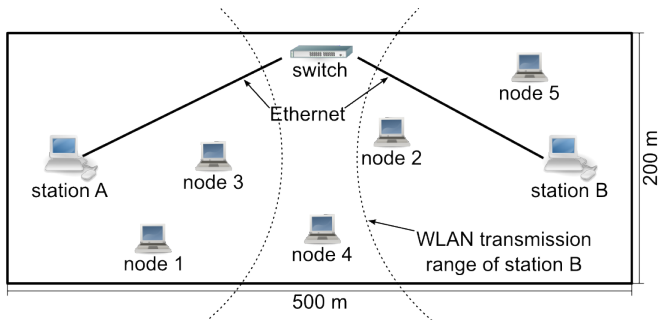
- 1 mobile temperature source (Celsius)
- 2 stationary temperature processors
- 2 mobile temperature sinks (Fahrenheit, Kelvin)

Methodology



- ▶ Simple very mobile scenario
 - ▶ 1 mobile temperature source (Celsius)
 - ▶ 2 stationary temperature processors
 - ▶ 2 mobile temperature sinks (Fahrenheit, Kelvin)

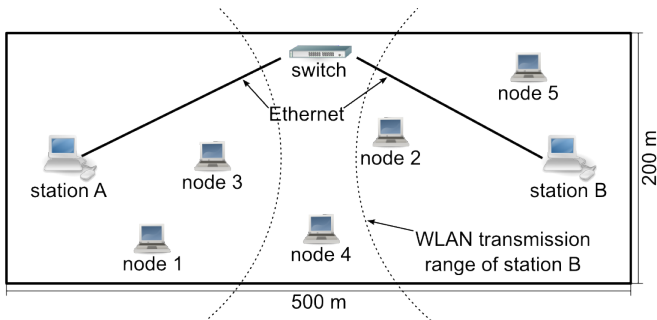
Methodology



► Scenario settings

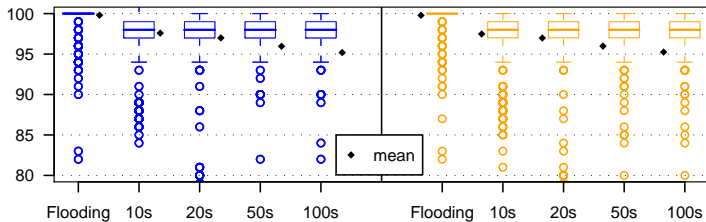
- Publication of 100 messages
- 1 message per second
- Flooding vs. ASP with differing announcement validity

Methodology



- ▶ Simulation environment
 - ▶ OMNeT++ version 3.3
 - ▶ INET Framework 20061020
 - ▶ 1000 runs per configuration

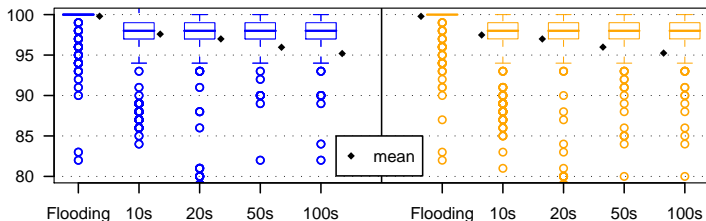
Completeness



- ▶ Flooding shows a 99.8% delivery rate
 - ▶ There are transmissions that can not be delivered
- ▶ Delivery rate of ASP drops with announcement validity



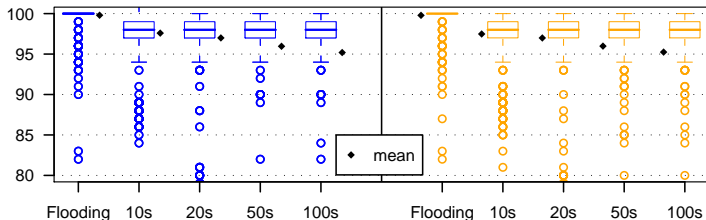
Completeness



- ▶ Flooding shows a 99.8% delivery rate
 - ▶ There are transmissions that can not be delivered
- ▶ Delivery rate of ASP drops with announcement validity



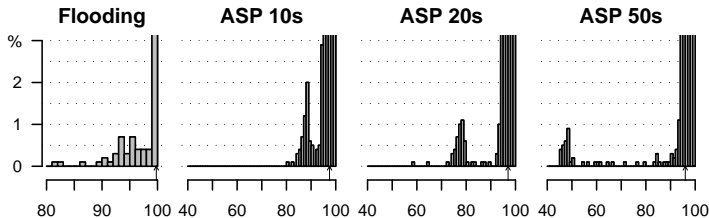
Completeness



- ▶ Flooding shows a 99.8% delivery rate
 - ▶ There are transmissions that can not be delivered
- ▶ Delivery rate of ASP drops with announcement validity



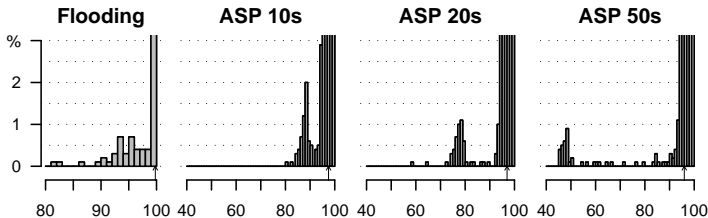
Completeness



- ▶ If a publication can not be delivered
 - ▶ One message is lost (no retransmission)
 - ▶ A new announcement is initiated
- ▶ If an announcement can not be delivered
 - ▶ No message is delivered for the validity period



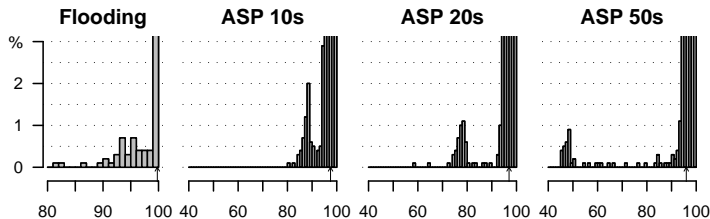
Completeness



- ▶ If a publication can not be delivered
 - ▶ One message is lost (no retransmission)
 - ▶ A new announcement is initiated
- ▶ If an announcement can not be delivered
 - ▶ No message is delivered for the validity period



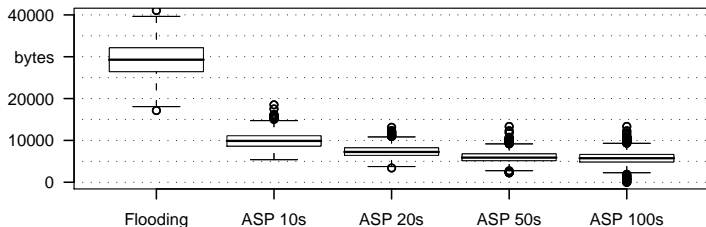
Completeness



- ▶ If a publication can not be delivered
 - ▶ One message is lost (no retransmission)
 - ▶ A new announcement is initiated
- ▶ If an announcement can not be delivered
 - ▶ No message is delivered for the validity period



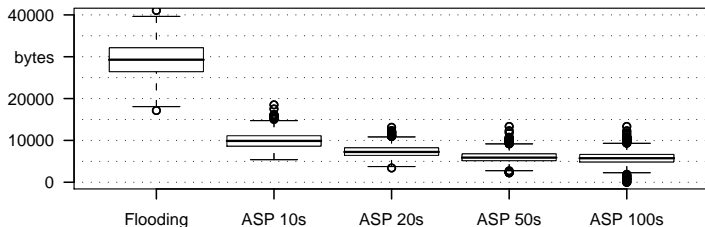
Network Load



- ▶ Load = data received from the network by NAL
- ▶ ASP generates from 33% to about 25% the load of Flooding



Network Load



- ▶ Load = data received from the network by NAL
- ▶ ASP generates from 33% to about 25% the load of Flooding



Outline

Introduction

Announcement/Subscription/Publication

System Architecture

Routing algorithm

Evaluation

Conclusion and Future Work



Conclusion

- ▶ **Benefits**
 - ▶ Independent of communication technology
 - ▶ Adaptable to high degree of mobility
 - ▶ Largely reduced load compared to flooding

- ▶ **Shortcomings**
 - ▶ Announcement validity problem
 - ▶ High -> less load but more messages are lost
 - ▶ Low -> less messages are lost but higher load
 - ▶ Decoupling in time only for message stream



Future Work

- ▶ Announcement caching
 - ▶ Overcome announcement validity problem
 - ▶ Allows for decoupling in time for each message
- ▶ Extension to and validation in other application classes
 - ▶ Large publications lead to
 - ▶ High resource usage
 - ▶ High network load
 - ▶ High message loss



Future Work

- ▶ Announcement caching
 - ▶ Overcome announcement validity problem
 - ▶ Allows for decoupling in time for each message
- ▶ Extension to and validation in other application classes
 - ▶ Large publications lead to
 - ▶ Message loss being crucial
 - ▶ High network load being a problem



Thank you for your attention

Henry Ristau*
University of Rostock
Information- and Communication Services Group
Rostock, Germany

*) Supported by a grant of the German National Research Foundation (DFG), Graduate School 1424, Multimodal Smart Appliance Ensembles for Mobile Applications (MuSAMA).

