

Mobil and ubiquitous computing

INF 5040 autumn 2007

lecturer: Frank Eliassen

Frank Eliassen, SRL & Ifi/UiO

1

Motivation

- Mobil computing is concerned with exploiting the connectedness of portable devices
- Ubiquitous computing is about exploiting the increasing integration of (small/tiny) computing devices in our everyday physical world
- Mobile and ubiquitous computing require particular solutions in many areas caused by dynamically changing computing environment: users, devices, and software components

Frank Eliassen, SRL & Ifi/UiO

2

Focus of this lecture

- Present open questions and research issues rather than solutions
- Student presentations will address some solution approaches
 - context-aware adaptation middlewares
 - middleware for wireless sensor networks

Frank Eliassen, SRL & Ifi/UiO

3

Open questions (some)

- How can software components associate and interoperate with one another while devices move, fail or spontaneously appear?
- How can systems become integrated with the physical world through sensing and context awareness?
- How to adapt to small devices' lack of computation and I/O resources?
- How to handle security in volatile, physically integrated systems?

Frank Eliassen, SRL & Ifi/UiO

4

Fields and subfields of mobile and ubiquitous computing

- Mobile computing
- Ubiquitous computing
- Wearable computing
- Context-aware computing

Frank Eliassen, SRL & Ifi/UiO

5

Volatile systems (1 of 2)

- Common system model for mobile and ubiquitous computing (and their subfields)
- Changes (or failures) are considered common rather than exceptional (in contrast to other types of systems where changes or failures are considered to be exceptions)
- Forms of volatility
 - failures of devices and communication links
 - changes in the characteristics of communication such as bandwidth
 - the creation and destruction of associations – logical communication relations – between software components resident on the devices
- Mobile and ubiquitous computing exhibit *all* of the above forms of volatility.

Frank Eliassen, SRL & Ifi/UiO

6

Volatile systems (2 of 2)

- Elements of the system model
 - smart spaces
 - device model
 - volatile connectivity
 - spontaneous interoperation

Frank Eliassen, SRL & Ifi/UiO

7

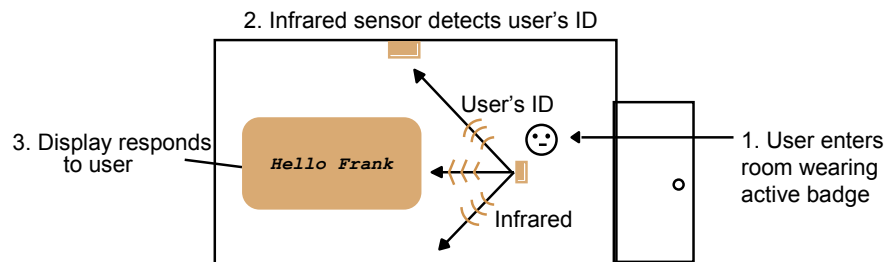
Smart spaces: environments within which volatile systems subsist

- A physical place/room with embedded services. The services are provided only or principally within that space
- Movements or 'appearance or disappearance' in a smart space:
 - Physical mobility
 - Logical mobilitet
 - Service/device appearance
 - Service/device disappearance
- GAIA: active spaces
 - <http://gaia.cs.uiuc.edu/>

Frank Eliassen, SRL & Ifi/UiO

8

Smart rooms that respond to a user with an active badge



Frank Eliassen, SRL & Ifi/UiO

9

Device model

- Limited energy
- Resource constraints
- Sensor and actuators
- Examples
 - Motes
 - Camera phones

Frank Eliassen, SRL & Ifi/UiO

10

Volatile connectivity

- Variation between different technologies (Bluetooth, WiFi, GPRS, etc)
 - Bandwidth, latency
 - Energy costs
 - Financial costs to communicate
- Disconnection
 - More likely in wireless networks
- Variable bandwidth and latency
 - Packet loss due to weak signal
 - Signal strength varies
 - Difficult to determine timeout-values in higher layer protocols due to varying conditions

Frank Eliassen, SRL & Ifi/UiO

11

Spontaneous interoperation

- In volatile systems, components routinely change the set of components they communicate with
 - take advantage of possibility to communicate with local components in a smart space, or a device may want to offer services to clients in its local environment
- **Association:** a logical relationship formed when at least one of a given pair of components communicates with the other over some well-defined period of time
 - Associations can be pre-configured, or spontaneous.
- **Interoperation:** interaction during an association

Frank Eliassen, SRL & Ifi/UiO

12

Pre-configured vs spontaneous associations: examples

Pre-configured	Spontaneous
Service-driven: <i>email client and server</i>	Human-driven: <i>web browser and web servers</i>
	Data-driven: <i>P2P file-sharing applications</i>
	Physically-driven: <i>mobile and ubiquitous systems</i>

Frank Eliassen, SRL & Ifi/UiO

13

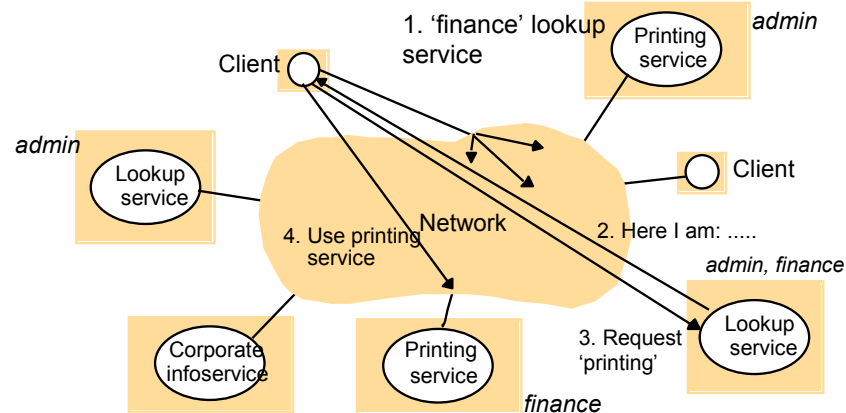
Association

- **Requirement:** a device that appears in a smart space needs to bootstrap itself in the smart space
- Two steps for bootstrapping itself:
 - Network bootstrapping (DHCP-server)
 - Establish associations between components on the device and services in the smart space
 - The association problem
 - How to constrain the scope to services in the smart space only (e.g., the hotel room)?
 - 'Boundary principle':
 - smart spaces need to have system boundaries that correspond accurately to meaningful spaces as they are normally defined (territorially or administratively)
- **Discovery-services:** one approach to the association problem (e.g., Jini: see Coulouris page 671)

Frank Eliassen, SRL & Ifi/UiO

14

Service discovery in Jini



Frank Eliassen, SRL & Ifi/UiO

15

Discovery services

- A directory service that is used to register and look up services in a smart space
- Requirements to discovery services
 - Service attributes is determined at runtime
 - Service discovery must be possible in a smart space without infrastructure to host a service discovery service
 - Registered services may spontaneously disappear
 - The protocols used for accessing the directory need to be sensitive to the energy and bandwidth they consume (cf. device model)

Frank Eliassen, SRL & Ifi/UiO

16

Interface to a discovery service

Methods for service de/registration	Explanation
<i>lease := register(address, attributes)</i>	Register the service at the given address with the given attributes; a lease is returned
<i>refresh(lease)</i>	Refresh the lease returned at registration
<i>deregister(lease)</i>	Remove the service record registered under the given lease
Method invoked to look up a service	
<i>serviceSet := query(attributeSpecification)</i>	Return a set of registered services whose attributes match the given specification

Frank Eliassen, SRL & Ifi/UiO

17

Design choices for discovery services

- Directory server or serverless
- Directory server: clients issue a multicast-request to locate the server (as in Jini)
 - Not all smart spaces have facilities for server implementations
- Serverless discovery: the participating devices collaborate to implement a distributed discovery service
 - **Push model:** servers multicast ('advertise') their descriptions regularly, and clients run their queries against them
 - **Pull model:** clients multicast their requests and devices providing matching services, respond
 - Both approaches are relatively resource demanding (battery, bandwidth) in their pure form
 - Exercise: how to improve the pull model?

Frank Eliassen, SRL & Ifi/UiO

18

Interoperation

- How can components that want to associate determine what protocol they can use to communicate?
- Main problem is incompatibility between software interfaces (components need not have been designed together)
- Two approaches:
 - Adapt interface to each other (interface adaptation): difficult
 - Constrain interfaces to be identical in syntax across as wide a class of components as possible
 - Example: Unix pipes (read, write)
 - Example: The set of methods defined in HTTP (GET, POST, ...)
 - Such systems are called ***data oriented***
 - Require additional mechanisms to describe type and value of data exchanged, such as the processing semantics of the server (difficult!)
 - **Data oriented programming models:** event-systems, tuple spaces, direct device interoperation (devices brought into direct association)

Frank Eliassen, SRL & Ifi/UiO

19

Sensing and context awareness

- How can systems be integrated with the physical world through sensing and context awareness?
- Sensing: use sensors to collect data about the environment
- Context aware systems: can respond to its (sensed) physical environments (location, heat, light intensity, device orientation, presence of a device, etc.) and the context can determine its (further) behaviour
- **Context** of an entity (person, place or thing): an aspect of its physical circumstances of relevance to system behaviour

Frank Eliassen, SRL & Ifi/UiO

20

Sensors

- Combination of hardware and software
- Sensors are the basis for determining contextual values
 - Location, velocity, orientation, ...
 - Temperature, light intensity, noise, ...
 - Presence of persons or things (e.g., based on RFID – electronic labels - or Active Badges)
- An important aspect of a sensor is its failure model
 - Some are simple (e.g., a thermometer often has known error bounds and distribution), some are complicated (e.g., accuracy of satellite navigation units depend on dynamic factors)

Frank Eliassen, SRL & Ifi/UiO

21

Sensor architectures

- Applications normally operate on more abstract values than sensors can produce
- Sensor abstractions important to avoid application level concerns with the peculiarities of individual sensors
- Therefore common to build a software architecture for sensor data as hierarchies
 - Nodes at a low hierarchical level provide sensor data at a low level of abstraction (longitude/latitude of a device)
 - Nodes at higher hierarchical levels (closer to the root node) provide sensor data at higher levels of abstraction (the device is in Frank's Cafe)
- Nodes at higher levels combine sensor data from lower levels both to abstract and to increase reliability

Frank Eliassen, SRL & Ifi/UiO

22

Context Toolkit: Example of sensor software architecture

- System architecture for general context aware applications
- Bases on 'context-widgets': reusable components that abstract over some types of context attributes (hide low level sensor details)
- Example: Interface to a IdentityPresence widget class

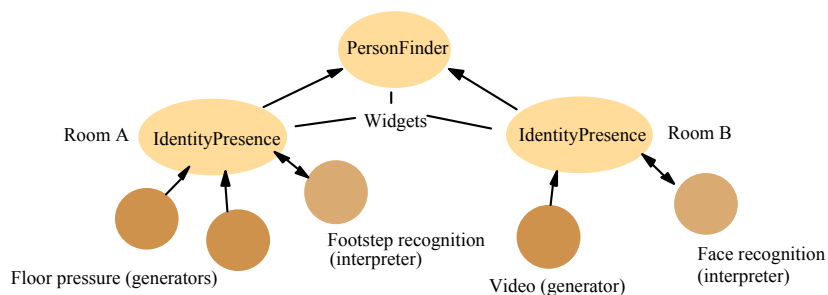
Attributes (accessible by polling)	Explanation
<i>Location</i>	Location the widget is monitoring
<i>Identity</i>	ID of the last user sensed
<i>Timestamp</i>	Time of the last arrival
Callbacks	
<i>PersonArrives(location, identity, timestamp)</i>	Triggered when a user arrives
<i>PersonLeaves(location, identity, timestamp)</i>	Triggered when a user leaves

Frank Eliassen, SRL & Ifi/UiO

23

Context Toolkit: Example of use of IdentityPresence widget

- A *PersonFinder* widget constructed by using IdentityPresence widgets ...



Frank Eliassen, SRL & Ifi/UiO

24

Wireless sensor networks (WSN)

- Network consisting of a (typically high) number of small, low-cost units or *nodes* that are more or less arbitrary arranged (e.g., "thrown out" in high numbers in a certain geographical area)
- Self-organising (ad-hoc network), functions independently of an infrastructure
- The nodes have sensing and processing capacity, can communicate wirelessly with a limited range (save energy), and act as routers for each other
- Are volatile systems because nodes can fail (battery exhaustion or otherwise destroyed (e.g., fire)), connectivity can change due to nodes failures

Frank Eliassen, SRL & Ifi/UiO

25

Three architectural features of wireless sensor networks

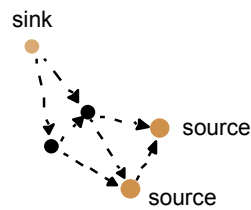
- Features driven by requirements of energy conservation and continuous operation
- *In-network processing*: The nodes have processing capabilities because processing is much less costly in energy consumption than (wireless) communication. Can be exploited to reduce the need for communication (only communicate when there is a need for it)
- *Disruption-tolerant networking*: based on store-and-forward transfer of data (not end-to-end)
- *Data oriented programming of nodes*: since nodes can fail, we can not rely on programming techniques for sensor nodes that refer to single nodes
 - *Directed diffusion*: a common programming technique that takes the above features into account

Frank Eliassen, SRL & Ifi/UiO

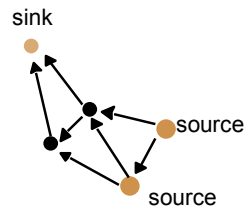
26

Directed diffusion

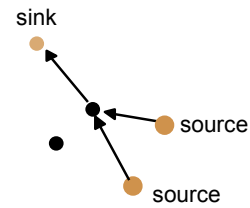
- Nodes (sources) have sensing capabilities (e.g., can measure temperature) or properties (e.g. location) that they can compare to needs for sensor information that they receive (as messages)
- Nodes that have a need for sensor information (sinks) declare this in "interest messages" that they send to neighbour nodes.



A. Interest propagation



B. Gradients set up



C. Data delivery

Frank Eliassen, SRL & Ifi/UiO

27

Adaptation

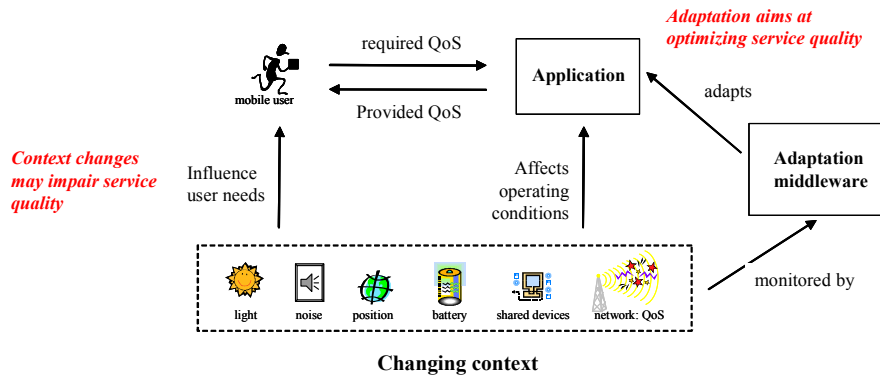
- Adapt the behaviour of an application to a dynamically varying context
 - Varying capabilities of different devices
 - Varying resource availability
 - User needs and wishes
 - Examples:
 - Dynamically adapt media quality (e.g., video) to available bandwidth and/or to user preferences, and/or to device capabilities (scaling and/or transcoding within same media type or between media types)
 - Dynamically adapt user interface to situation of user or the orientation of the device ...
- Media content providers can not in advance know about all different devices now or in the future
=> requires generally dynamic approach to adaptation

Frank Eliassen, SRL & Ifi/UiO

28

Example: MADAM middleware

- Integrates context management and adaptation
 - <http://www.ist-madam.org>
 - <http://www.simula.no/departments/networks/projects/MADAM/>



Frank Eliassen, SRL & Ifi/UiO

29

Announcing INF5360: Seminar on Dependable and Adaptive Distributed Systems

- First time: Spring 2008 (Tuesdays 10:00 – 13:00)
- Seminar content
 - Explores state-of-the art principles, methods, and techniques for devising adaptive and dependable distributed systems.
- The seminars covers
 - Architectural and infrastructural principles for adaptive and dependable distributed systems.
 - Adaptivity and dependability in service-oriented architectures, grid computing, P2P systems, mobile and wireless environments.
 - Approaches to improve the scalability of dependable and adaptive systems.
 - Evaluation and experience reports on dependable and adaptive distributed systems and services.
- Builds on INF5040

Frank Eliassen, SRL & Ifi/UiO

30

Summary

- Most challenges to mobile and ubiquitous systems are caused by their volatile nature
- In such environments applications need to be context aware and adaptive
 - Integrated with the physical world through sensing and context awareness
 - Adapt to changes in the physical circumstances by changing behavior
- There are many challenges, but yet only few (comprehensive) solutions