Ubiquitous Computing
Context and Context-Awareness

University of Innsbruck
WS 2010/2011
Context as Filter

Context is a **filter** striking against the everyday information overload.
What is Context? What does Context-Aware mean?

- In the early 90's, primarily position information were considered as contextual parameters (Schilit et al., Want et al., Leonhardt, etc.)

- Examples of more generic definitions of context which consider broader range of parameters:

- Pascoe, 1998:
  
  "Context is the subset of physical or conceptual states of interest to a particular entity"

  "Context-awareness is the ability of a program or device to sense various states of its environment and itself"
What is Context? What does Context-Aware mean?

- Schmidt, 1999:
  "A context describes a situation and the environment a device or user is in. A context is identified by a unique name. For each context, a set of features is relevant. For each relevant feature a range of values is determined (implicit or explicit) by the context"

- Chen and Kotz, 2000:
  "Context is the set of environmental states and settings that either determines an application's behaviour or in which an application event occurs and is interesting to the user"
What is Context? What does Context-Aware mean?

Most popular definition of context and context-awareness:

**Definition of Context:**
“Any information that can be used to characterize the situation of an entity. An Entity is a person, place or object that is considered relevant […] concerning service usage.”  
(A.K. Dey: “Understanding and using context”, 2001)

**Definition of Context Awareness:**
“A system is context-aware if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user’s task.”  
(A.K. Dey: “Understanding and using context”, 2001)
Discussion

- Definition of Dey and Schmidt et al. are both incomplete but complement each other:
  - Schmidt et al. require range for features
  - Dey is more focused on characterization of an entity
  - Disadvantage in both definitions is the dependency to the term situation
  - Important component is the relevance
- Pascoe as well as Chen and Kotz emphasize on states.
- In all definitions: Restriction to user/device or user/service interaction. Why not e.g. service/service?
Relevance

- Relevance independent context
  - Situation is modelled and tracked for any kind of aspect (dimension) irrespective whether the state is important for current interaction or not

- Relevance dependent context
  - Situation is modelled for any kind of aspect, but contextual information is derived and applied just if it is relevant for the current interaction

- Some refer to the first as situation awareness and as context awareness to the latter. Some just don’t care.

1) of a user, a vehicle, the environment or any other entity
Relevance

- In this lecture we shall see the context as relevance dependent:
  - „A context is the set of all for a specific task relevant context information“
  - „A situation is the set of all known context information“

(Strang: “Service-Interoperability in UbiComp Environments”, 2004)
Context vs. Content

- What is the difference between context and content?
  - All context-aware services will work w/o context (e.g. in case of too high uncertainty or temporary lack of data), but not w/o content
  - Context is used to optimize an interaction, meaning that context is a „better“ content (personalized, localized, etc.)
  - Context shall be used to minimize interaction with the user or the service
Primary Dimensions of Context

- Location
- Time
- Identity
- Activity

being part of almost all context models
Location vs. Context

- “There is more to context than location“ [Schmidt et al. 1999]
- For example:
  - Network connectivity, battery capacity
  - Environment: presence of physical devices (e.g. printers), displays, other persons/objects
  - Light, Noise, Temperature, Pressure, Humidity
  - Social Situation, Mood
- Basically, anything that can be sensed can be modelled as context information, even many things where no sensor is (yet) invented (e.g. mood)
Incompleteness / Uncertainty

- Context used to approximate state and user intent
- Example: Museum tour guide infers from proximity between person and exhibit an interest of the person in the exhibit, but in fact the person may just coincidently stopped near the exhibit, doing some conversation. Additional context would be necessary to fully understand the situation, which will never be sufficient for 100% certainty. Thus any context-aware system must inherently support uncertainty, i.e. express certainty in their beliefs and use the context information in dependency to the (low or high) level of certainty.
Ambiguity

- Many sources of information are many sources for errors – incorrect sensor readings, incorrect inferences, divergent interpretation of data, to name just a few reasons.

- Serious context-aware systems must include the existence of *ambiguities* in their model and provide solutions for ambiguity resolution strategies.

- One typical approach dealing with context ambiguity is to combine multiple disparate sources of context, i.e. a *sensor fusion* approach such as Hidden Markov Modeling.
"Context-awareness is one of the key drivers of the ubiquitous computing paradigm, whereas a well designed context modelling and retrieval approach is a key accessor to the context in any context-aware system."

[Strang and Linnhoff-Popien, 2004]
Context Modelling Approaches

- **Key-Value-Pairs Models**
  - most simple category of models
  - not very efficient for more sophisticated structuring purposes
  - only exact matching
  - no inheritance

*Environment Variables: Key-Value-Pairs*
Context Modelling Approaches

- Markup Scheme Models
  - scheme implements model
  - typical representatives: *profiles*
- Examples:
  - Extensions of
    - Composite Capabilities/Preference Profile (CC/PP)
    - User Agent Profile (UAProf)
  - Comprehensive Structured Context Profiles (CSCP)
  - Pervasive Profile Description Language (PPDL)
  - Centaurus Capability Markup Language (CCML)

```
<?xml version="1.0"?>
<rdf:RDF
  xmlns:rdf = "http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:cscp = "http://example.org/CSCPProfileSyntax#"
  xmlns = "http://example.org/SessionProfileSyntax#"
  xmlns:dev = "http://example.org/DeviceProfileSyntax#"
  xmlns:net = "http://example.org/NetworkProfileSyntax#">
  <SessionProfile rdf:ID="Session">
    <cscp:defaults rdf:resource="/context/CSCPProfile/previous#Session" />
    <device rdf:DeviceProfile="dev">dev:DeviceProfile" dev:Devic">
      <dev:Hardware>dev:Hardware</dev:Hardware>
      <dev:Memory>9216</dev:Memory>
    </dev:DeviceProfile>
  </SessionProfile>
</rdf:RDF>
```

CSCP Instance based on RDF
Context Modelling Approaches

- **Graphical Models**
  - particularly useful for structuring, but usually not used on instance level
  - **Examples:**
    - Well known: *UML*
    - *Contextual Extended ORM*

![Graphical Models Diagram]

*Contextual Extended ORM*
Context Modelling Approaches

- Logic Based Models
  - Logic defines conditions on which a concluding expression or fact may be derived from a set of other expressions or facts (reasoning)
  - context is defined as facts, expressions and rules
- High degree of formality
- Examples:
  - McCarthy’s *Formalizing Context*
  - Akman&Surav’s *Extended Situation Theory*

\[
\begin{align*}
S_1 &= [s_1 | s_2 \vdash \langle \text{bird}, \hat{a}, 1 \rangle] \\
S_2 &= [s_1 | s_2 \vdash \langle \text{flies}, \hat{a}, 1 \rangle] \\
B &\vdash \langle \text{present}, \hat{a}, 1 \rangle \land \langle \text{penguin}, \hat{a}, 0 \rangle \land \ldots \\
C &= S_1 \Rightarrow S_2 \mid B
\end{align*}
\]

*Context Expression from Extended Situation Theory*
Object Oriented Models

- Intention behind object orientation is (as always) encapsulation and reusability

Examples:

- *Cues* (TEA project)
- *Active Object Model* (GUIDE project)
Context Modelling Approaches

- Ontology Based Models
  - Ontology used as *explicit* specification of a shared conceptualization → strong in the field of normalization and formaliy
  - Context is modelled as concepts and facts
  - Examples:
    - CoBrA system
    - ASC model of Context Ontology Language (CoOL)
    - CONON
The ASC Context Modelling Approach

"An ontology is a hierarchically structured set of terms for describing a domain that can be used as a skeletal foundation for a knowledge base."

by Swartout, Patil, Knight and Russ, 1996

Important distinguishing feature: Ontologies are **property oriented**.

Father/son conversation:

"Dad, is a ferrari a red car with a little horse on it?"
"That’s correct, son, why?"
"I think it is passing us just now!"
Property orientation allows for „fuzzy“ context reasoning!
Context Modelling Approaches

- **Probabilistic Models**
  - Everything is modeled as possible worlds which have a certain probability
  - Inherently good to model uncertainty with probability distributions (PDF)
  - Dependencies are modeled as networks
  - Examples:
    - Bayesian Networks
Context Retrieval

- Context retrieval (e.g. through inferencing) is the act of making sense of input data from sensors and other sources, to determine the context of an entity.

- Once this context has been inferred, the application/service can take an appropriate action (at least likely more appropriate than based on content).

- This usually brings up additional issues such as how to resolve ambiguity or uncertainty in context.
## Context Retrieval

<table>
<thead>
<tr>
<th>Modeling Approach</th>
<th>Standard Retrieval Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key-Value-Pairs Models</td>
<td>Linear Search</td>
</tr>
<tr>
<td>Markup Scheme Models</td>
<td>Markup Query Language</td>
</tr>
<tr>
<td>Graphical Models</td>
<td>Transformation</td>
</tr>
<tr>
<td>Logic Based Models</td>
<td>Inferencing</td>
</tr>
<tr>
<td>Object Oriented Models</td>
<td>Algorithm</td>
</tr>
<tr>
<td>Ontology Based Models</td>
<td>Reasoning</td>
</tr>
<tr>
<td>Probability Models</td>
<td>Calculation</td>
</tr>
</tbody>
</table>
Appropriateness Criteria for context modelling in UbiComp

- Distributed Composition
- Partial Validation
- Richness and Quality of Information
- Incompleteness and Ambiguity
- Level of Formality
- Applicability to existing environments

[Strang and Linnhoff-Popien, 2004]
Requirement 1: Distributed Composition

- Any ubiquitous computing system is a derivative of a distributed computing system which lacks of a central instance being responsible for the creation, deployment and maintenance of data and services, in particular context descriptions. Instead, composition and administration of a context model and its data varies with notably high dynamics in terms of time, network topology and source.
Requirement 2: Partial Validation

- It is highly desirable to be able to partially validate contextual knowledge on structure as well as on instance level against a context model in use even if there is no single place or point in time where the contextual knowledge is available on one node as a result of distributed composition. This is particularly important because of the complexity of contextual interrelationships, which make any modeling intention error-prone.
Requirement 3: Richness and Quality of Information

- The quality of a information delivered by a sensor varies over time, as well as the richness of information provided by different kinds of sensors characterizing an entity in an ubiquitous computing environment may differ.
Requirement 4: Incompleteness & Ambiguity

The set of contextual information available at any point in time characterizing relevant entities in ubiquitous computing environments is usually incomplete and/or ambiguous, in particular if this information is gathered from sensor networks. This should be covered by the model, for instance by interpolation of incomplete data on the instance level.

some stories don't have a clear beginning, middle, and end. Life is about not knowing, having to change, taking the moment and making the best of it, without knowing what's going to happen next. delicious ambiguity...

-gilda radner
Requirement 5: Level of Formality

- It is always a challenge to describe contextual facts and interrelationships in a precise and traceable manner. For instance, to perform the task “print document on printer near to me”, it is required to have a precise definition of terms used in the task, for instance what “near” means to “me”. It is highly desirable, that each participating party in an ubiquitous computing interaction shares the same interpretation of the data exchanged and the meaning “behind” it (so called shared understanding), and that this can be formally proved.
Requirement 6: Applicability to existing environments

From the implementation perspective it is important that a context model must be applicable within the existing infrastructure of ubiquitous computing environments, e.g. a service framework such as Web Services.
## Model Appropriateness for UbiComp Environments

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Distributed Composition</th>
<th>Partial Verification</th>
<th>Richness &amp; Quality of Information</th>
<th>Incompleteness &amp; Ambiguity</th>
<th>Level of Formality</th>
<th>Applicability to service frameworks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key-Value Models</td>
<td>-</td>
<td>-</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>+</td>
</tr>
<tr>
<td>Markup Scheme Models</td>
<td>++</td>
<td>++</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>++</td>
</tr>
<tr>
<td>Graphical Models</td>
<td>--</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Object Oriented Models</td>
<td>++</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Logic Based Models</td>
<td>++</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>++</td>
<td>-</td>
</tr>
<tr>
<td>Ontology Based Models</td>
<td>++</td>
<td>++</td>
<td>+</td>
<td>+</td>
<td>++</td>
<td>+</td>
</tr>
<tr>
<td>Probability Models</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>++</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
Other important arguments

- How about traceability?
  - Sometimes called scrutability
  - Component of trust, which is itself a concept of the security&privacy domain, but also an important aspect of social interaction
Service Usage Evolution Chain

- Distributed Service Usage
  - Mobile Networks
  - Mobile Information Access
  - Adaptive Applications

- Mobile Service Usage
  - Context-Awareness
  - Ad-hoc Networks
  - Smart Sensors & Devices

- Ubiquitous Service Usage

Two main benefits from Context-Awareness:
- Adaptation to changes in environment without user interaction
- Effective information filter (typical mobile devices have limited UI!)

Location-Awareness is a special kind of Context-Awareness.

Modeling & Integration Requirements:
- high level of formality
- distributed composition
- partial validation
- incompleteness
- quality of information
- applicability to existing service frameworks

Lecture Ubiquitous Computing, Prof. Dr. Thomas Strang, WS 2010/2011
Context-Aware Service Example

1. Click!
2. Uploading pics
3. Pickup photos
Excurse: MNM service model

Contextual Extended MNM service model

[Diagram of service provider domain, context provider domain, and customer domain with relationships and roles such as "uses", "manages", "realizes", "supplies", "accesses", "provides", "directs", "substantiates", "manages", and "concludes" connections between service, service provider, context provider, and customer.]
Context-Aware Web Service Interaction

Service Directory (e.g. SLP-DA)

Direct Service Discovery

Announcements

SOAP Interaction

Customer Domain

Delegated Svc. Discovery

incl. Context Binding Lookup

Invoc. Reasoner (Inference Engine)

Service Provider Domain

Middleware/Intermediate

Context Informationen

Relevance Criteria

"Pickup-Station near my Hotel"

"Photo Geiger, 87629 Füssen"

Context Provider Domain

Rules

Ontologies

Facts

Monitor & Event Generator

ASC

CoOL based Knowledge

Query

Knowledge

Context Mgt. Implementation

Context Provider Domain

Rule & Event Generator
Relevance as Applied Reasoning
(here: F-Logic)

Relevance-Criteria:
“Consider all pickup-stations as relevant which are less than 5km from Neuschwanstein/Füssen”.

FORALL Shop, Position <- EXISTS Distance, ImpScale, intDistance
   Position:#WGS84ContextInformation[#characterizes->Shop:#Entity] AND
   ImpScale[#useOperation->#SpatialDistanceFromSymbolInterOp@("Füssen")] AND
   Distance:#IntegerContextInformation[#integerValue->intDistance; #usedByScale->ImpScale:#MeterScale; #characterizes->Position] AND
   lessorequal(intDistance, 5000).

Inferencing Result:
0:#PhotoPorst,
  #Schwangau
0:#KaisersDrugstore,
  #Fuessen

Customer Domain

Service Provider Domain

Reasoner (Inference Engine)

shopVector = pickupStations();
pickupStations( shopVector );

Middleware
Context-Aware Semantic Web Service: The Food Delivery Example (1/2)

Three types of parameter:

• input parameter
• output parameter
• implicit parameter

Which values expects the operation?

Which values delivers the operation?

For which values is the service itself available?

Scope „business hours“ for service „food delivery“

Lecture Ubiquitous Computing, Prof. Dr. Thomas Strang, WS 2010/2011
Context-Aware Semantic Web Service: The Food Delivery Example (2/2)

Scope „business hours“

User

NCS Handover

motivates

implicit parameter „current time“
Contemporary Context-Aware Applications

- Typical, almost **canonical examples** of context-aware applications
  - Tour guide
  - Restaurant/ATM finder
  - Message/call handling, often combined with interruptability in the office and while mobile
  - Health care / ambient assisted living