Principles of Context aware systems

Presented by: Rajesh Gangam
DEFINITIONS
Why?

Why to study context?

- User
  - Take advantage of Human-Computer dialogue.
  - Increased Richness in Human-Computer Interaction ➔ More useful computational services.

- Developer
  - Determine what Context Behaviors to support in their applications.
  - How to choose the Context in the Application.
Contents - 1:

- Context
- Categories of Context
- Context aware computing
- Characterization
Context:

- “Context is 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** to the **interaction** between a user and an application, including the user and applications themselves.
Categories of Context

- Why?
  - Help Application Designers uncover the most likely pieces of context that will be useful in their applications.

- Primary Context types are
  - Location
  - Identity
  - Activity
  - Time

- Secondary Context types
Context Aware Systems

“\textit{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.}”

\textbf{Common Misunderstood is Self Modifying Systems.}
Categorization of Features for Context Aware Applications

- **Presentation** of Information and services to the user.
  - Example: Display

- **Automatic** execution of a service
  - Example: Alarm

- **Tagging** of context to information for later retrieval.
  - Example: Prediction
### Table 1. Application of context and context-aware categories

<table>
<thead>
<tr>
<th>System Name</th>
<th>System Description</th>
<th>A</th>
<th>I</th>
<th>L</th>
<th>T</th>
<th>Context-Aware</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cyberguide [1]</td>
<td>Tour guide</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Stick-e Documents [3,4,5]</td>
<td>Tour guide</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>Paging and reminders</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reactive Room [6]</td>
<td>Intelligent control of audiovisuals</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>CyberDesk [8,9,10]</td>
<td>Automatic integration of user services</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Responsive Office [12]</td>
<td>Office environment control</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>NETMAN [13,16]</td>
<td>Network maintenance</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Fieldwork [17,18,22]</td>
<td>Fieldwork data collection</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Augment-able Reality [19]</td>
<td>Virtual post-it notes</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Context Toolkit [24]</td>
<td>In/Out Board</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>Capture of serendipitous meetings</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Active Badge [28]</td>
<td>Call forwarding</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>
Contents - 2

- Common Architecture Principles
- Middleware/ Layered Conceptual framework
- Context Models
- Context Abstractions
- Context Aware Frameworks
- Requirements for dealing with context.
- Context Toolkit
SOFTWARE FRAMEWORKS AND DESIGN
Frameworks

- Direct Sensor Access
- Middleware Infrastructure
  - A Layered Architecture.
- Context Server

More..

- Widgets
- Networked Services
  - Can thwart unreliability.
- Blackboard Model
  - Publish-Subscribe Model.
Middleware/ Layered Conceptual Framework

- Application
- Storage/Management
- Preprocessing
- Raw data Retrieval
- Sensors

- Synchronous
- Asynchronous
- Extraction
- Aggregation
- Physical Sensors
- Virtual Sensor
- Logical Sensors

Raw data

Middleware/ Layered Conceptual Framework

Usable Security – CS 6204 – Fall, 2009 – Dennis Kafura – Virginia Tech
Context Models

- Key-Value Models
- Markup Scheme Models
- Graphical Models
- Object oriented Models
- Logic based Models
- Ontology based Models
  - Simplicity, flexibility and extensibility, genericity, expressiveness
  - Context type, Context Value, Time Stamp, Source, Confidence.
Context Abstractions

- Context Widgets
  - Provides Reusability, inheritance and encapsulation.
- Interpreters
  - Interpretation from sensory data.
- Aggregators
  - Similar to Widget. Differs in how context is acquired and used.
- Services
  - Synchronous and Asynchronous.
- Discoverers
  - Similar to ARP and RARP in computer Networks.
Context Aware Frameworks

- Architectures
- Resource Discovery
- Sensing
- Context Model
- Context Processing
- Historical Context Data
- Security and Privacy
Requirements for Dealing with Context

- Separation of context.
- Context Interpretation
- Transparent, Distributed Communication
- Constant availability of Context Acquisition
Features in Context Toolkit

- Distributed Communications
- Subscriptions
- Event Handling
- Discovery
- Context Services
<table>
<thead>
<tr>
<th>Architecture</th>
<th>Sensing</th>
<th>Context model</th>
<th>Context processing</th>
<th>Resource discovery</th>
<th>Historical context data</th>
<th>Security and privacy</th>
</tr>
</thead>
<tbody>
<tr>
<td>CASS</td>
<td>Centralised middleware</td>
<td>Sensor nodes</td>
<td>Relational data model</td>
<td>Inference engine and knowledge base</td>
<td>n.a.</td>
<td>Available</td>
</tr>
<tr>
<td>CoBra</td>
<td>Agent based</td>
<td>Context acquisition module</td>
<td>Ontologies (OWL)</td>
<td>Inference engine and knowledge base</td>
<td>n.a.</td>
<td>Available</td>
</tr>
<tr>
<td>Context Toolkit</td>
<td>Widget based</td>
<td>Context widgets</td>
<td>Attribute-value tuples</td>
<td>Context interpretation and aggregation</td>
<td>Discoverer component</td>
<td>Available</td>
</tr>
<tr>
<td>CORTEX</td>
<td>Sentient object model</td>
<td>Context component framework</td>
<td>Relational data model</td>
<td>Service discovery framework</td>
<td>Resource management component framework</td>
<td>Available</td>
</tr>
<tr>
<td>Gaia</td>
<td>MVC (extended)</td>
<td>Context providers</td>
<td>4-ary predicates (DAML + OIL)</td>
<td>Context-service module (first-order logic)</td>
<td>Discovery service</td>
<td>Available</td>
</tr>
<tr>
<td>Hydrogen</td>
<td>Three layered architecture for various context types</td>
<td>Adapters for various context types</td>
<td>Object-oriented</td>
<td>Interpretation and aggregation of raw data only</td>
<td>n.a.</td>
<td>n.a.</td>
</tr>
<tr>
<td>SOCAM</td>
<td>Distributed with centralised server</td>
<td>Context providers</td>
<td>Ontologies (OWL)</td>
<td>Context reasoning engine</td>
<td>Service locating service</td>
<td>Available</td>
</tr>
</tbody>
</table>
Discussion

- Reliability of the Sensors is not considered in most of the Context Aware Designs.
- What changes will the Context Aware Programs need while addressing Automatic Trust Negotiation and Policy managements.
References:


Thank You!