Context-aware toolkits

Zalia Shams

Overview

- The Context Toolkit:
  - Aids development of Context-Enabled applications
- An Architectural Framework : AURA
  - Supports task migration
  - Resource adaptation

The Context Toolkit: Aiding the Development of Context-Enabled Applications

Daniel Salber,
Anind K. Dey,
Gregory D. Abowd

GVU Center, College of Computing
Georgia Institute of Technology

http://www.cc.gatech.edu/fce/ctk/

The Context Toolkit:
Aiding the Development of Context-Enabled Applications

Daniel Salber, Anind K. Dey and Gregory D. Abowd
GVU Center, College of Computing
Georgia Institute of Technology
Atlanta, GA 30332-0280
+1 404 894 7512
{salber, anind, abowd}@cc.gatech.edu

ABSTRACT
Context-enabled applications are just emerging and promise richer interaction by taking environmental context into account. However, they are difficult to build due to their distributed nature and the use of unconventional sensors. The concepts of toolkits and widget libraries in graphical user interfaces has been tremendously successful, allowing programmers to leverage off existing building blocks to build interactive systems more easily. We introduce the concept of context widgets that mediate between the environment and the application in the same way graphical widgets mediate between the user and the application. We illustrate the concept of context widgets with the beginnings of a widget library we have developed for sensing presence, identity and activity of people and things. We assess the success of our approach with two example context-enabled applications we have built and an existing application to which we have added context-sensing capabilities.

Keywords
Context-enabled or context-aware computing, ubiquitous computing, toolkits, widgets, applications development

INTRODUCTION
Over the last decade, several researchers have built applications that take advantage of environmental information, also called context, to enhance the interaction with the user. The construction of these context-enabled applications is cumbersome, and currently no tools are available to facilitate the development of this class of applications. This paper presents a toolkit for developing reusable solutions for handling context information in interactive applications.

We first define the notion of context, and through a brief review of the literature identify the key challenges of developing applications that sense context, followed by an overview of the paper.

What Is Context?
Environmental information or context covers information that is part of an application's operating environment and that can be sensed by the application. This typically includes the location, identity, activity and state of people, groups and objects. Context may also be related to places or the computing environment. Places such as buildings and rooms can be fitted with sensors that provide measurements of physical variables such as temperature or lighting. Finally, an application may sense its software and hardware environment to detect, for example, the capabilities of nearby resources.

Sensing context information makes several kinds of context-enabled applications possible: Applications may display context information, capture it for later access and provide context-based retrieval of stored information. Of major interest are context-aware applications, which sense context information and modify their behavior accordingly without explicit user intervention.

Why Use Context?
Usage scenarios of typical context-enabled applications found in the literature have led us to identify recurrent challenges, which we further detail below.

Mobile tour guides are designed to familiarize a visitor with a new area. They sense the user's location and provide information relevant to both the user and the location she's at [1, 3, 6, 10]. Likewise, office awareness systems sense users' locations, but are also interested in their activities to help people locate each other, maintain awareness or forward phone calls [12, 17, 18]. In ubiquitous computing systems, devices sense and take advantage of nearby resources: a handheld computer located next to an electronic whiteboard may make use of the larger display surface or allow the user to interact with other nearby handheld users [14, 18]. Finally, context-based retrieval applications gather and store context information and allow later information retrieval based on context information. For instance the user can ask a note-taking application to pull up the notes taken at a previous meeting with the group she's meeting with currently [9, 13].

Outline

- Purpose
- Contribution
- Context Widgets
  - Component
  - Implementation
- Example Widgets
- Summery

Purpose

- Extension of the concepts of toolkits and widget libraries in Graphical User Interfaces (GUI) into context widgets.

- Leverage off existing building blocks to build interactive systems that mediate between user and applications.

Contribution

- Illustrates the concept of context widgets.
- Developed a widget library.
- Applied the widgets into an existing application.
- Developed two context-enabled applications with the context-widgets.

Context

- Context
  - "Environmental information or context covers information that is part of an application's operating environment and that can be sensed by the application."
  - Includes
    - location,
    - identity,
    - activity and
    - state of people, group and objects.

Context Widget

- Live in a distributed architecture.
- Monitor environmental information all the time.
- Context widgets have a state and a behavior
  - State: Context information.(e.g. arrival time of a person).
  - Behavior: Triggers callbacks when environment change is detected (e.g. new person arrives).
- Benefits
- **Hide the complexity** of the actual sensors from the application(s)
- **Abstract context information** to suit the expected needs of applications (e.g., filters detail)
- Provide **reusable building blocks** of context sensing (like GUI widgets)

Context Widget

- Composition: Widgets may consist of any number of three components
  - Generator: Acquires raw context information (e.g. sensor data for user ID)
  - Interpreter: Abstracts that information (e.g. Actual user name)
  - Server: Aggregate information as a gateway between applications.

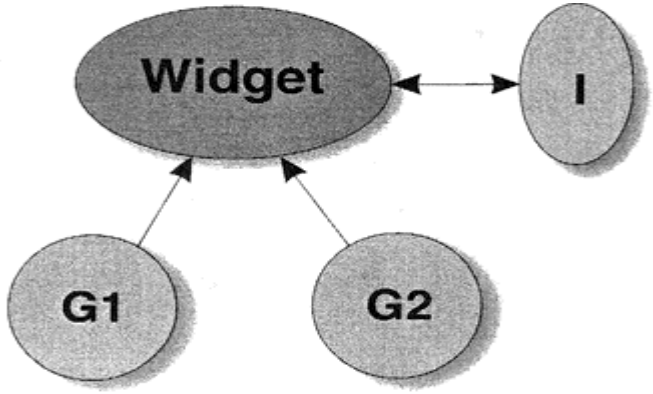


Figure: The widget gets raw data from two generators G1 and G2 and requests the services of an interpreter I.

Context Widget

- Communication Across Heterogeneous Components:
  - Communication model uses HTTP protocol.
  - Language model uses the ASCII-based Extensible Markup Language (XML).
- Privacy Concerns:
  - A server can be used to encapsulate a privacy manager for its given domain.
  - Personal server can implement privacy restrictions.

Identity Presence Widget

- Reports arrival and departure of people with their identity and time at the place of it's specified location.

Widget Class	Identity Presence
Attributes	
Location Identity Timestamp	Location the widget is monitoring ID of the last user sensed Time of the last arrival
Callbacks	
PersonArrives (location, identity, timestamp) PersonLeaves (location, identity, timestamp)	Triggered when a user arrives Triggered when a user leaves

Current implementation

- Dallas Semiconductor's **ibutton.**
- Users snap it in a reader to notify their presence.
- Each button has a unique ID from which the user's identity is derived.

Activity Widget

- Senses current activity level at a location such as a room.

Widget Class	Activity
Attribute	
Location Timestamp AveragedLevel	Location the widget is monitoring Time of the last change in activity level Activity Level (none, some, a lot) Average over a user-specified time interval
Callbacks	
ActivityChange(location,AveragedLevel, timestamp)	Triggered when the activity level changes from one level to another

Current implementation uses microphone.

Some Other Widgets

- NamePresence Widget
  - Similar to IdentityPresence widget but provides user's actual name instead of ID.
- PhoneUse Widget
  - Provides information whether a phone is being used and length of use.
- MachineUse Widget
  - Provides information about when a user logs onto or off of a computer, his identity and length of his computing
- GroupURLPresense Widget
  - Provides a URL relevant to the research group user belongs to when her presence is detected.

In/Out Board

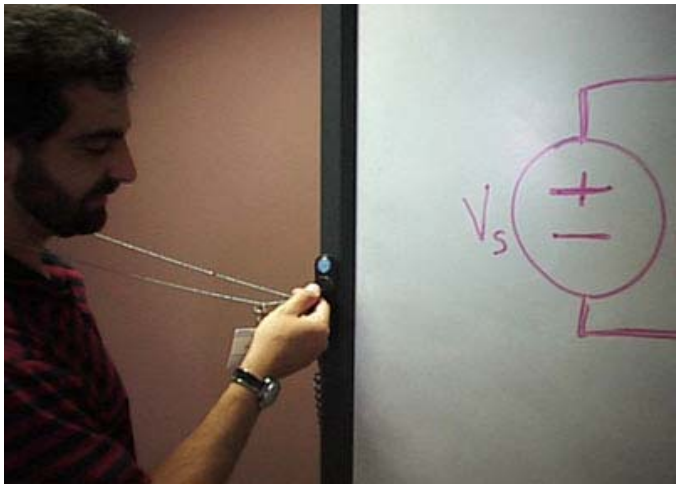
- The board is used to indicate which members of the office are currently in the building and which are not.
- Context Information:
  - a participant's identity and
  - the time at which they arrived or departed.
- Widget:
  - IdentityPresence widget located at the entrance.

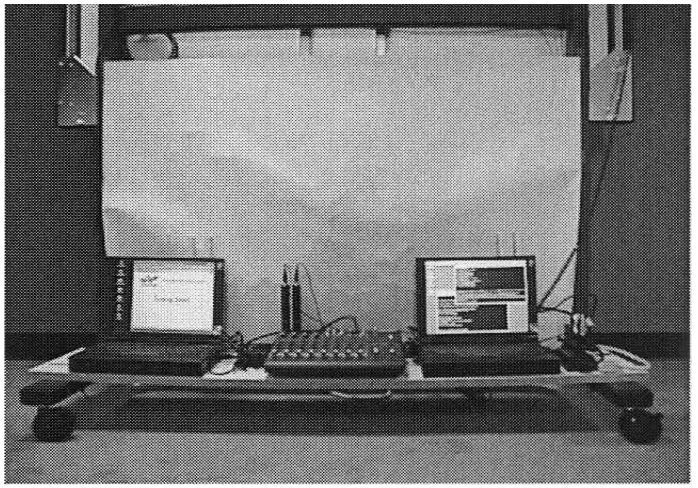
Information Display

- Displays information relevant to the user's location and identity on a display adjacent to the user.
- It activates itself as someone approaches it, and the information it displays changes to match the user, her research group, and location.
- Context Information:
  - Location of the display,
  - The identity of the user,
  - The research group the user belongs to and
  - Any particular information the research group needs.
- Widget: Uses GroupURLPresence Widget.

DUMMBO(Dynamic Ubiquitous Mobile Meeting) Meeting Board

- An instrumented digitizing whiteboard to support meetings
- Captures audio and whiteboard drawings.
- The context information:
  - the participants' identities,
  - the time of when they arrived at or left the mobile whiteboard, and
  - The location of the mobile whiteboard.
  - Uses multiple *NamePresence widgets.*





Conclusion

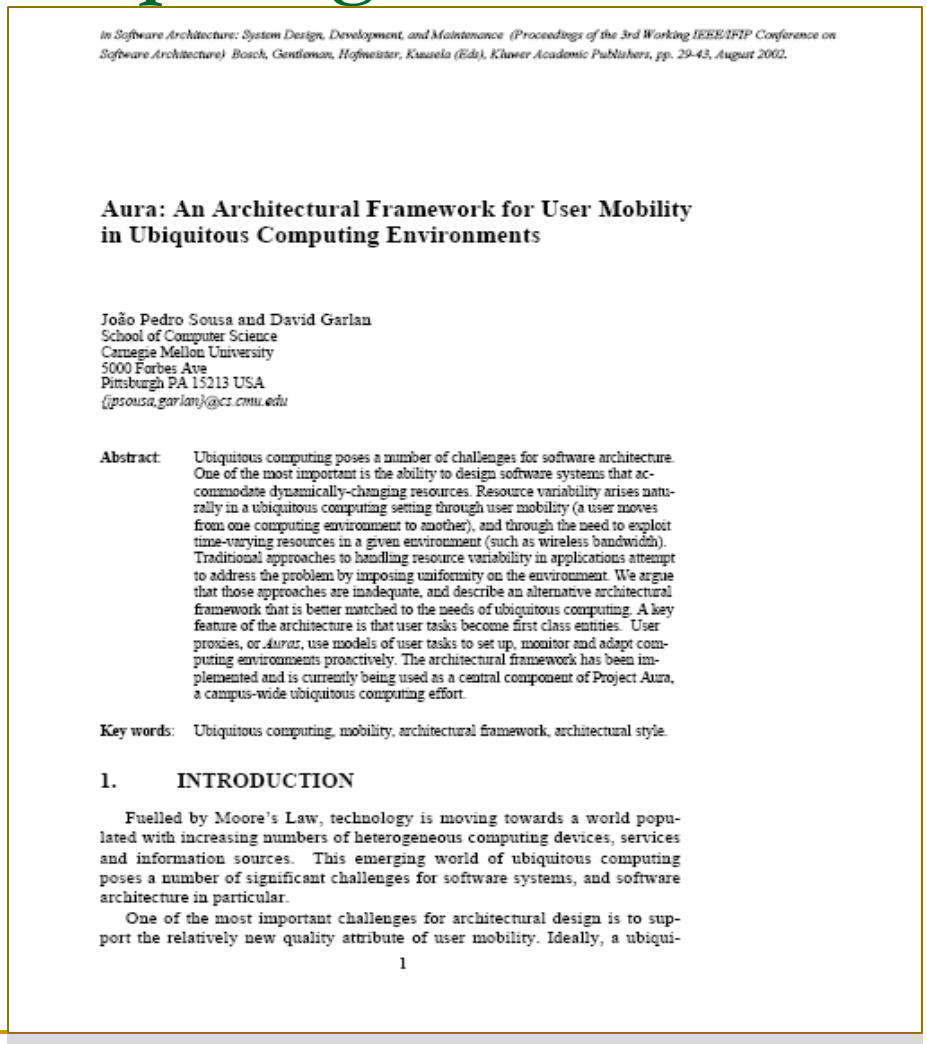
- Introduced context toolkit that supports the development of context-enabled applications.
- Separates concerns between context sensing and application semantics.
- Applied context widgets in In/Out board, DUMMBO and Information display applications.

Aura: An Architectural Framework for User Mobility in Ubiquitous Computing Environments

João Pedro Sousa
David Garlan

School Of Computer Science
Carnegie Mellon University

http://www-2.cs.cmu.edu/~aura/



in Software Architecture: System Design, Development, and Maintenance (Proceedings of the 3rd Working IEEE/IFIP Conference on Software Architecture) Bosch, Gentleman, Hofmeister, Kuusela (Eds), Kluwer Academic Publishers, pp. 29-43, August 2002.

Aura: An Architectural Framework for User Mobility in Ubiquitous Computing Environments

João Pedro Sousa and David Garlan
School of Computer Science
Carnegie Mellon University
5000 Forbes Ave
Pittsburgh PA 15213 USA
{jpsousa,garlan}@cs.cmu.edu

Abstract: Ubiquitous computing poses a number of challenges for software architecture. One of the most important is the ability to design software systems that accommodate dynamically-changing resources. Resource variability arises naturally in a ubiquitous computing setting through user mobility (a user moves from one computing environment to another), and through the need to exploit time-varying resources in a given environment (such as wireless bandwidth). Traditional approaches to handling resource variability in applications attempt to address the problem by imposing uniformity on the environment. We argue that those approaches are inadequate, and describe an alternative architectural framework that is better matched to the needs of ubiquitous computing. A key feature of the architecture is that user tasks become first class entities. User proxies, or *Auras*, use models of user tasks to set up, monitor and adapt computing environments proactively. The architectural framework has been implemented and is currently being used as a central component of Project Aura, a campus-wide ubiquitous computing effort.

Key words: Ubiquitous computing, mobility, architectural framework, architectural style.

1. INTRODUCTION

Fuelled by Moore's Law, technology is moving towards a world populated with increasing numbers of heterogeneous computing devices, services and information sources. This emerging world of ubiquitous computing poses a number of significant challenges for software systems, and software architecture in particular.

One of the most important challenges for architectural design is to support the relatively new quality attribute of user mobility. Ideally, a ubiqui-

1

Outline

- Purpose
- Contribution
- Features
- Components
- Example Scenario
- Conclusion

User Mobility

Fred is leaving for his office

Purpose

- Support of user mobility is one of the most important challenges for architectural design.
- Current Architectural frameworks assumes homogeneous computing and does not support
  - Diverse capabilities of each environment
  - Dynamic variation of capabilities and recourses in the environment.
- Today's systems distract a user in many explicit and implicit ways, thereby reducing his effectiveness.

Contribution

- Finds out inadequacy of present architectural designs for supporting ubiquitous computing.
- Presented a new architectural framework able to fulfill needs of ubiquitous computing more than traditional designs.
- Implemented the architectural framework as central component of project AURA.

Key features of the architectural framework

- Users tasks are represented
  - as coalitions of abstract services.
  - explicitly and autonomously from a specific environment.

- Environments are equipped to self-monitor and renegotiate task support to handle dynamic variation of capabilities and resources.

Personal Aura

- Personal Aura acts as a proxy for mobile user.
- It marshals appropriate resources for corresponding user at the time of his entering into a new environment.(e.g. for text editing Microsoft word or Emacs).
- Captures constraints that the physical context around the user imposes on tasks. (e.g. bandwidth).

Component of Architecture Framework

- Task Manager (Prism)
- Context Observer
- Environment Manager
- Supplier of Services

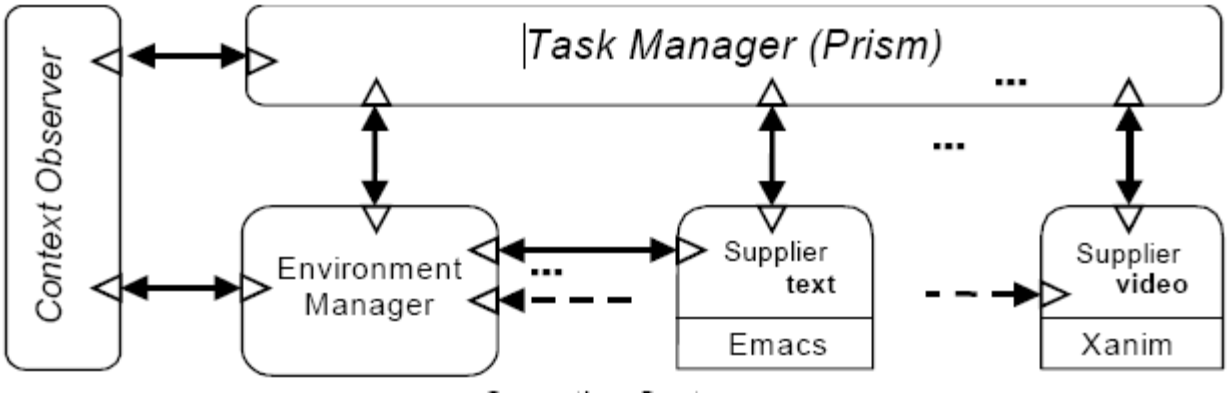
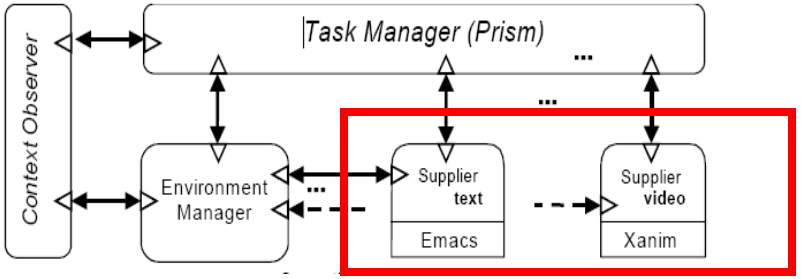


Figure: Aura Bird's eye view

Service Suppliers

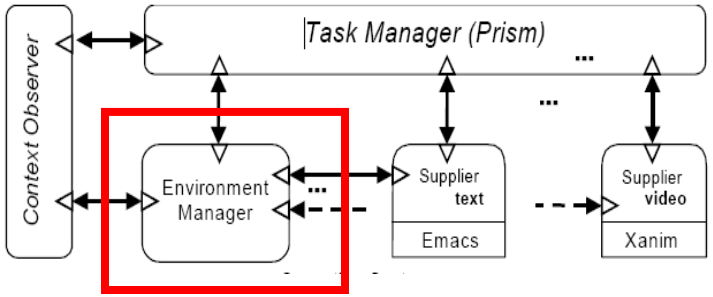
- Provide abstract services that tasks are composed of.
- Implemented by wrapping existing applications and services to conform to Aura API's.



- Example:
  - Microsoft Word and Notepad can be wrapped as supplier of text editing services.

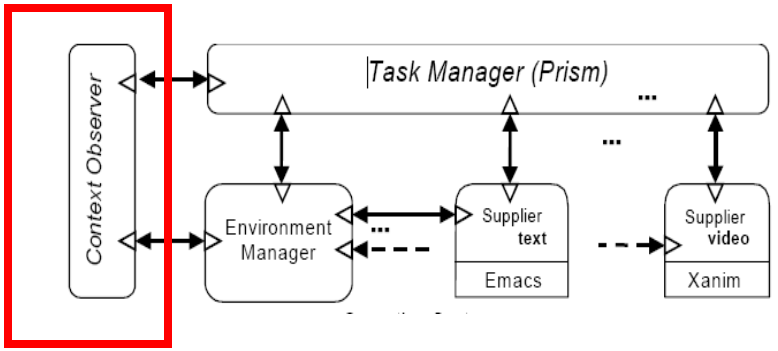
Environment Manager

- Embodies the gateway to the environment.
- Suppliers are registered in with local environment manager.
- Environment manager selects service supplier matching better to user's preferences on request for services.



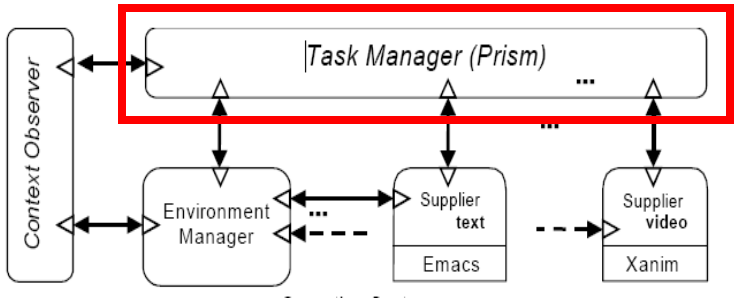
Context Observer

- Provides information on physical context.
- Reports relevant events in the physical context back to Task Manger and Environment Manager.
- Depending on the sensors deployed in the environment may have different degree of sophistication.



Task Manager (Prism)

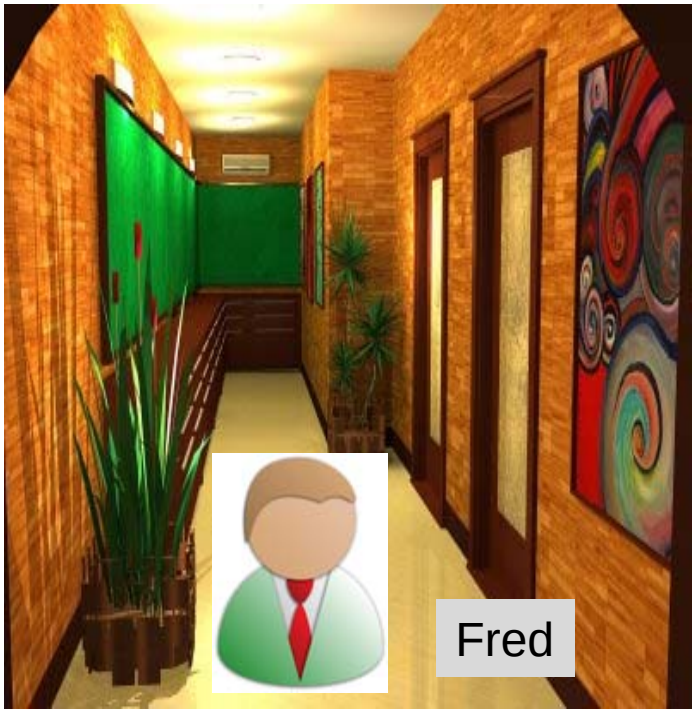
- Minimize user distractions in four changes:
  - User moves to another environment
    - Migrates all task information to new environment
    - Negotiates task support with new environment manager
  - Environment Change
    - Monitors quality of service
    - Queries Environment manager for alternative configuration if needed.
  - Task Change
    - Saves the state of interrupted task
    - Instantiated new task
  - Context change
    - Adjusts or suspends parts of task affected by context change



AURA at WORK

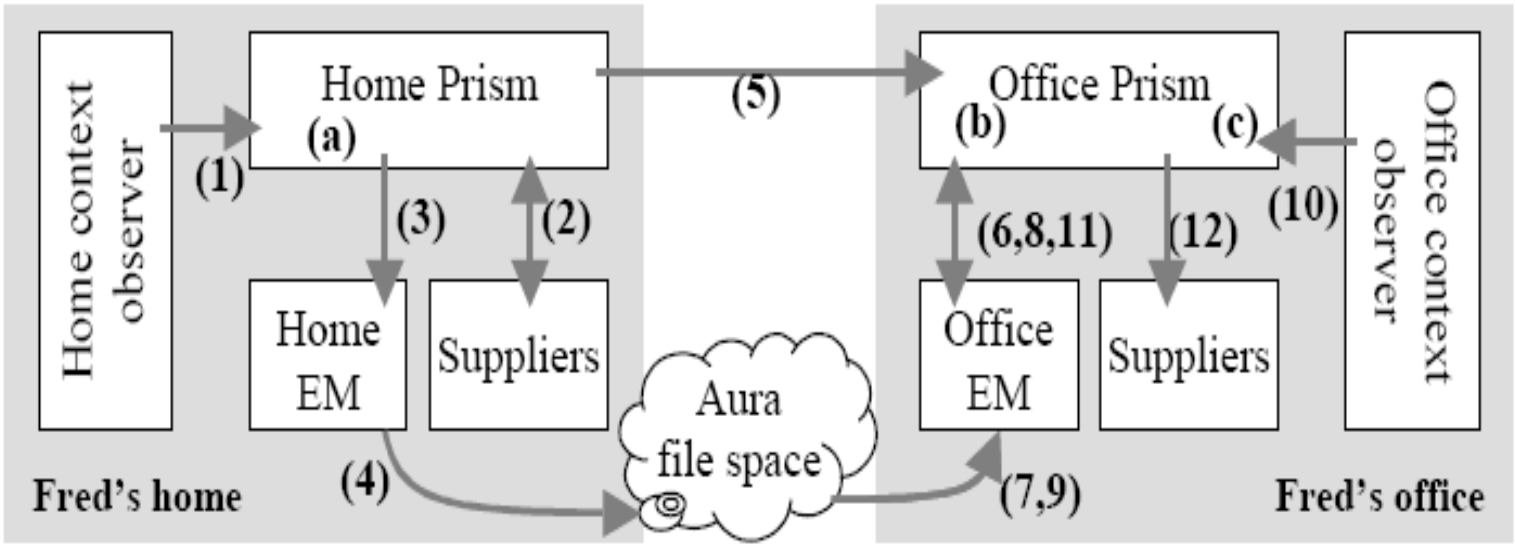
Fred is leaving for his office

AURA at WORK



Fred is at his office

AURA at WORK



Fred goes from home to the office

Conclusion

- Presented an architectural framework.
- The framework allows:
  - A user to preserve continuity of his/her work when moving between environments.
  - An ongoing task to adapt with dynamic resource variability.
- Capabilities encapsulated in components.
- Prototype has been implemented.

Discussion

- Do you think AURA framework handles personal preference and intention appropriately?
- “Traditional approaches to handling resource variability in applications attempt to address the problem by imposing uniformity on the environment”- Do you think AURA framework actually differ from that? If so, why?
- What kind of security problem may remain in AURA framework?
- Context toolkit supports environment data change. Do you think only environmental data change is sufficient to use these widgets in real world?
- What other dynamism can be added to context toolkit?