"Prottoy": A Context Aware Application Framework

Fahim Kawsar, Kaori Fujinami, and Tatsuo Nakajima

Department of Information and Computer Science, Waseda University, Japan

{fahim, fujinami, tatsuo}@dcl.info.waseda.ac.jp

ABSTRACT

This paper presents a hypothetical framework concept ("Prottoy") focusing on generalizing the interactions of context aware applications with underlying smart environment. The paper also focuses on the issues related to generalization considered in "Prottoy".

Keywords

Context Awareness, Generalization

INTRODUCTION

Heterogeneity is one of the main characteristics of Ubiquitous Computing Environments. One of the major challenges of ubicomp applications is to provide solution that meets this challenge with a satisfactory level. This leads to the application developer to deal with numerous devices with numerous varying properties and features like accuracy, precision, location, position etc. We believe the future environment will be aware of its operating context and will be augmented with many sentient objects for ease of our daily life. For application development encountering these smart environment, populated with numerous varying artifacts we need a software infrastructure that generalizes these sentient artifacts access to some extend and provides a generic interface for seamless development of application in context aware environment.

We are investigating this generalization notion, to what level generalization can be provided in context aware environment and identifying where is the boundary, and based on our findings we are working on the development of a software infrastructure "Prottoy". This paper presents the hypothetical concept of the initial prototype of the framework and our findings so far.

RELATED WORK

Currently there exist a number of context aware application frameworks in the literature. The Sentient Computing Project [1] utilizes Active Bat location system to provide architectural base for indoor application. HP Cool Town [2] encapsulates the world by providing web presence of place, people and thing and allows interaction with web presence of entities primarily exploiting RF technology. EasyLiving [3] focuses on an architecture that supports coherent user experience as users interact with variety of devices in a smart environment. Open Agent architecture [4] is an agent-based system, which exploits a centralized black board to support contextual behavior. Schilits System Architecture [5] deals with context awareness by Device Agents that maintain status and capabilities of devices, User Agents that maintain user preferences and Active Maps that maintain the location information of devices and users. Context Toolkit [6] focuses on component abstraction by providing notion of Context Widget and Context Aggregator for sensor devices and entities with multiple contextual information. Discoverer manages these components and additionally there is a Context Interpreter component that performs the task of context interpretation.

However, none of these frameworks focuses on achieving truly generalization. Many of them have some interesting features that "Prottoy" exploits in its architecture. The following table presents a comparison of feature supports of some of these and proposed architectures.

CS: Context Specification SEP: Separation of Concerns

TDC: Transparent Distributed Communication IN: Interpretation

CA: Constant Availability ST: Storage RD: Resource Discovery

GEN: Context Generalization SEC: Security Concerns

P= Partial Support √= Full Support X= No Support

System	CS	SEP	TDC	IN	CA	ST	RD	GEN	SEC
Open Agent Architecture	P	√	√	√	X	X	X	X	X
Easy Living	P	√	√	P	X	X	P	X	X
Schililt's System	P	P	P	X	√	X	P	X	X
Context Toolkit	√	√	√	√	√	√	√	X	X
Prottoy	√	√	√	X	√	√	√	√	√

Table 1: Comparison of Feature Support

PROPOSED FRAMEWORK DESIGN

We believe the term generalization is very hard to achieve in ubiquitous environment. As a consequence, rather than proposing a concrete infrastructure and building application on top of it, we are building the infrastructure in an iterative manner based on our experiences and findings on the development of applications with varying user requirement in varying environment. The initial prototype of our framework composed of the following components:

- Resource manager: Keeps track of all the resources
- Widget Wrapper: Encapsulates hardwired widgets and devices and control access to these widgets. We call sensor augmented everyday artifacts and similar service providers as hardwired widget. Context Toolkit use similar component notion as Context Widget
- Context Integrator: Integrates multiple contexts
- Virtual Widget: Generalizes Access Interface for the widget wrappers
- Context Analyzer: Analyzes the acquired context
- Context Storage: Stores Context for future retrieval
- Preference Component: Manages user preferences and privacy.

There is no context interpreter in our framework. Since providing generalization is the primary concerns of this framework, the task of context interpretation has been omitted as context interpretation is application dependent

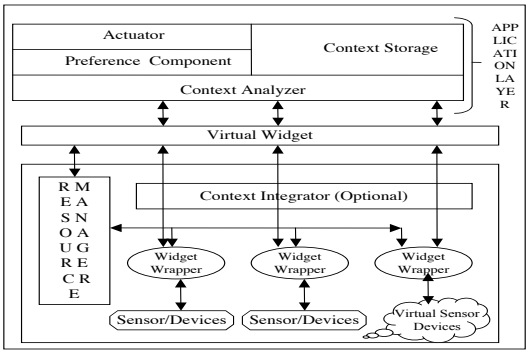

Fig 1: "Prottoy" Framework Design

and cannot be generalized in broad sense. We expect application developer to provide the application logic at the application layer, using the context analyzer that offers the contextual information either by analyzing after retrieving from virtual widget or predicting by utilizing context storage. The underlying widgets join the environment dynamically and their properties and features are dynamically binded by the resource manager, which later exploited by virtual widget. We separate the application layer from the sentient environment by providing the virtual widget notion. This encapsulates the low level sensors augmented artifacts and provides application a generic interface to interact with them. For application development using Prottoy, developers need not to interact with low level sensor augmented devices. For using any artifacts the developer has to create an instance of generic Virtual Widget like

```
VirtualWidget sentientObj = new
VirtualWidget(Context,Location,Accuracy,CommOpt)
```

And later use its interfaces to manipulate the low level artifacts in the same manner regardless of their varying properties and features like

```
objValue=sentientObj.getValue();
```

Obviously we are working to provide interfaces for all sort of property query and registration support during dynamic binding and later retrieval by application in a generalized way. We hope to provide more detail implementation notion in larger version of the paper.

DEMO APPLICATION

We have developed a demo simple media level application "Smart Assistant" using "Prottoy" architecture consisting of multiple sentient artifacts namely Sentient Chair, Sentient Dish Tray and Sentient Lamp. Sentient Chair is aware of its state of usage, whereas Sentient Dish Tray knows what is put on it, and their interaction history, finally Sentient Lamp can identify the presence of user and can automatically be turned on/off based on light level and user presence. Combining the information retrieved from these artifacts, the application can monitor user activity and based on its analysis it suggest user for refreshment, asks user about refreshment and can control workspace's lighting. Current implementation can talk to user and can make few gestures on the display to get user attention. We have found that application development is fairly easy with "Prottoy" for interacting with such environment as application developer can focus concretely on the application logic considering each artifact as generic object having same interfaces and the framework handles the rest.

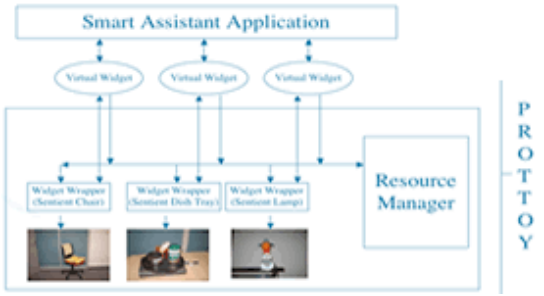

Fig 2: Application's interaction with "Prottoy"

FINDINGS AND FUTURE WORK

Sentient devices exhibit varying properties like owner, accuracy, precision, location, position etc. As a whole it is important to identify these device properties for interacting with them in advance, our current implementation provides very thin interface for such property registration and acquisition. Additionally we have to provide both publishing and subscription mechanism for state acquisition along with polling mechanism. Another important issue is, how artifacts should join the environment, should they register themselves to a global resource manager or they themselves act as a local resource manager? Also we need to provide suitable procedure for controlling artifact access and user level preference management. We are working on these issues to identify what may be the best answers to these questions to support generalization in context aware environment and hope to incorporate them in "Prottoy" accordingly.

CONCLUSION

"Prottoy" concentrates on generalizing the context aware environment for application development. It promises to be a strong software base as we hope to provide the necessary features to achieve the highest level of generalization possible.

REFERENCES

1. A. Harter at el. The anatomy of a context-aware application. *In Mobile Computing and Networking*
2. Caswell at el. Creating Web representations for Places *Proceedings of the 2nd International Symposium on Handheld and Ubiquitous Computing*
3. Brumitt. at el EasyLiving: Technologies for Intelligent Environments. *In the Proceedings of the 2nd International Symposium on Handheld and Ubiquitous Computing (HUC2K),*
4. Cohen at el An Open Agent Architecture. *In the Proceedings of the AAAI Spring Symposium Series on Software Agents*
5. Schilit, Bill N. System architecture for context-aware mobile computing. PhD dissertation.
6. A. Dey at el A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Human-Computer Interaction*