Using Internet Services to Manage Massive Evolving Information for Ubiquitous Computing Systems

Emre Kıcıman
Stanford University
emrek@cs.stanford.edu

ABSTRACT

A major challenge in building ubiquitous computing systems is the large variety of heterogeneous devices. Building applications that cope with this heterogeneity requires managing massive amounts of quickly evolving information, mapping among the various semantically-equivalent functionalities of devices. We advocate using Internet services to store and collect these mappings. We are implementing this hypothesis in the IMHome project, focusing on the spontaneous interaction of media creation, storage, and playback devices. To evaluate IMHome, we are developing metrics to measure the dependability of IMHome-based applications.

Keywords

Internet services, Ubiquitous computing, Massive evolving information, Heterogeneity, Dependability

INTRODUCTION AND MOTIVATION

One of the major challenges in building a ubiquitous computing (ubicomp) system is ensuring that it can scale to the large degree of heterogeneity present in ubicomp environments [3, 1]. The difficulty is that an application must explicitly know how to interoperate with the massive variety of devices in real-world ubicomp environments. Even more challenging, new devices are being developed every day.

For example, consider a ubiquitous computing application that plays the first thirty seconds of every song in a user's music collection—perhaps as part of a "name that tune" game. This is not complicated functionality and seems quite easy to build—except that this application must be able to control everything from CD players to MP3 players to tomorrow's next-generation audio player, all with slightly different capabilities and potentially significantly different control interfaces. And while it is easy to build an *ad hoc* application for any specific stereo system, it is practically impossible to extend the same *ad hoc* application to support an open-ended and evolving variety of audio players. The application simply has to know too much.

MASSIVE EVOLVING INFORMATION

This heterogeneity problem is a member of a small but important class of ubicomp challenges we call *massive evolving information* (MEI) problems. We characterize this class of problems with two properties:

1. For any specific, static scenario, a simple *ad hoc* solution exists, based on *a priori* knowledge about the scenario.
2. A solution in a general, dynamic environment is theoretically achievable. Practically however, the information required is too massive and/or changes too rapidly to be feasibly stored and managed in a local ubicomp environment.

Other MEI problems include associating annotations (such as product reviews) with physical artifacts, and mapping machine-level identifiers to human-understandable names.

To create a general solution to an MEI problem, we must either reduce the amount of information that must be known (such as through standardization) or we must find a way to store and manage large, dynamic data sets in ubicomp environments. Standardization efforts have been made for years, but have so far failed to stabilize due to changing requirements and technologies. And though we can store increasingly large data sets in ubicomp environments, we still have issues distributing updates at the scale required by many MEI problems.

USING INTERNET SERVICES TO MANAGE MEI

In contrast to our ability to store and update massive evolving information in ubicomp environments, there are well-known techniques for scaling Internet services to manage large amounts of data, large numbers of queries, and rapid updates. Today's Internet services routinely manage massive amounts of data.[1]

By building an Internet service to manage our massive and evolving information and accessing it remotely from our ubicomp environments, we reduce the particular MEI problem to a domain-specific issue of how to generate the large data set and keep it up-to-date. Though this is potentially still a difficult issue, it is generally more tractable.

One example that illustrates how Internet services can be used to solve MEI problems is found in today's Internet CD databases.[2] The problem of linking a music CD to its album information and related music is an MEI problem, since 1) it is trivial to build a system which stores related information for a static set of albums; and 2) the rapid rate at which new CDs are published makes it difficult to build a local system with knowledge of all CDs.

[1] At the extreme, the Internet Archive (archive.org) manages over 100TB of data. Major search engines routinely manage multiple terabytes of data.

[2] Popular versions of this service include http://www.CDDB.com and http://www.FreeDB.org/

A partial solution is to embed this information in the CD itself, but that precludes linking, say, an artist's first releases to their later albums. Instead, the solution used today is to use an always-accessible Internet service to store and manage information related to CDs, and distribute the work of entering new CD information across the users of the service.

RESEARCH HYPOTHESIS

We hypothesize that by using Internet services to manage massive, evolving data sets, we can build a useful solution for solving the semantic heterogeneity problem in ubiquitous computing. Specifically, we propose to use an always-accessible Internet service to manage approximate-equivalence mappings among the functionalities of and control commands for various heterogeneous devices.

Using Internet services to logically centralize the storage of this mapping information leaves us with two subproblems. First, to use approximate-equivalence mappings among devices' functionalities, there must be agreement on the mechanics of accessing those functions. Current trends in self-describing data formats and communication protocols such as XML and tuplespaces are providing these standard mechanisms. Additionally, their clean separation of syntax from semantics makes it easier to apply mediation techniques to transform among formats [2].

Secondly, as new devices are built and new classes of functionalities are developed, we must update the equivalence mappings in our Internet service. The simplest method for doing this is to provide the end-user of the system with the option of manually providing a mapping when one is missing. To help willing users determine the correct mappings, we can provide them with human-language descriptions of devices and functionalities as extra context. Once a new mapping is created, it can be pushed back to our Internet service to be used by others.

IMHOME

We are testing our hypothesis in the context of IMHome, a project focusing spontaneous interoperation of devices for media creation, storage and playback in the home. Relevant details of the IMHome architecture include its use of a *task abstraction* to organize cooperating devices and services; a simple, self-describing *property-based control interfaces* for devices and tasks; and *property mapping services*, deployed as Internet services, to programmatically transform the control interfaces of devices and tasks.

EVALUATION

The problem of evaluating systems software in ubiquitous computing environments extends well beyond the scope of this thesis summary. However, since our community has not yet settled on specific evaluation methods, we believe it is worth discussing.

Though researchers in the systems software community have traditionally concentrated on the performance aspects of systems, their focus is now moving towards dependability [4]. This new focus has special significance for ubicomp, where a dynamic and fragile environment poses unique challenges to building robust systems [5].

We believe that an important one-axis for measuring systems software contributions to ubicomp is whether they improve the dependability of ubicomp applications. Contributions which enable new functionality should ideally improve dependability as well; any contribution that compromises dependability should be met with skepticism.

By enabling applications to more robustly adapt to ubicomp's heterogeneous environment, we expect our framework to improve the overall dependability of ubicomp applications. To verify this hypothesis, we are developing metrics for measuring the dependability of IMHome applications based on its task and control abstractions.

SUMMARY AND FUTURE RESEARCH

The goal of our research is to provide a useful solution for spontaneously using heterogeneous devices together. We are focusing our efforts on media devices in the home environment and are completing a first prototype of the IMHome system. We are currently integrating multimedia devices into this system and developing sample applications. We have recognized that the management of massive evolving information is key to practically solving issues of semantic heterogeneity, and are beginning to build a prototype Internet service for managing the equivalence-mappings of IMHome device functionalities. We plan to evaluate our system based on how much it improves the dependability of ubicomp applications.

ACKNOWLEDGEMENTS

I would like to thank Atsushi Shionozaki for his feedback on the IMHome project. This work has been supported by a Stanford Graduate Fellowship, an NSF Graduate Fellowship, and a gift from the Sony Corporation.

REFERENCES

1. W. Edwards and R. Grinter, At Home with Ubiquitous Computing: Seven Challenges. In *Third Intl. Conference on Ubiquitous Computing (Ubicomp2001)*, 2001.

2. E. Kıcıman and A. Fox, Using Dynamic Mediation to Integrate COTS Entities in a Ubiquitous Computing Environment. Proceedings of *The Second International Symposium on Handheld and Ubiquitous Computing 2000.*

3. T. Kindberg and A. Fox, System Software for Ubiquitous Computing. *IEEE Pervasive Computing Magazine* 1(1):7081, January 2002.

4. D. Patterson, et al. Recovery Oriented Computing: Motivation, Definition, Principles, and Examples. *UC Berkeley Computer Science Technical Report* UCB//CSD-02-1175, March 15, 2002.

5. S. Ponnekanti, B. Johanson, E. Kıcıman and A. Fox. Designing for Maintainability, Failure Resilience, and Evolvability in Ubiquitous Computing Software. In Submission to *Operating Systems Design and Implementation 2002.*