

Small, Rich, and Adaptable

Robert Hirschfeld^{*}, Katsuya Kawamura^{*}, and Jeff Eastman^{**}

^{*}DoCoMo Euro-Labs, Munich, Germany

^{**}Windward Solutions, Los Altos, California

hirschfeld@acm.org

In the world of pervasive computing, a set highly distributed software entities is working together to concurrently fulfill a wide range of tasks on behalf of multiple users. These entities are running on interconnected computing nodes that vary widely in power and configuration. They are made aware of their computational and physical environment via sensors and can affect their environment via actuators. While some of the devices, sensors and actuators may be easily accessible physically, most of them are assumed to be field deployed and thus only remotely accessible. The set of entities forms a continuously running system that needs to adapt to a changing environment as well as changing user requirements imposed on it.

Driven by low cost and low power consumption, most approaches to pervasive computing require the utilization of hardware with rather restricted computational resources. Due to this fact, higher-level languages associated with resource demanding platforms are often ruled out from the beginning. This leaves to developers with only very low-level programming languages such as C or even assembly to choose from. Unfortunately such programming languages offer neither a adequate level of abstraction to appropriately model complex domain concepts nor do they provide enough run time flexibility to adapt running entities to new environmental conditions or new user requirements.

We propose a dynamic object environment for ubiquitous computing to be based on the following technologies (Figure 1):

- Late binding
- Runtime reflection
- Small runtimes

Late binding is key to effectively addressing change at runtime by allowing us to defer some choices to a later point in time. This avoids premature commitments to design decisions. This is especially important when sensors and actuators are pluggable or need to be reconfigured at run time. In contrast, early binding requires us to provide abstractions addressing all possible changes and configurations at a very early point in time. This often leads to incorrect or inadequate abstractions.

Systems with reflective architectures incorporate structures representing aspects of themselves [4]. The aggregate of these structures is called a system's self representation allowing the system to both observe its own computation (introspection) as well as influence or change it (intercession). For system adaptation, introspection will allow us to observe computational properties of a deployed set of software entities as well as the

computational environments they are running in. Intercession can be based on our observations and result in the alteration of such software entities.

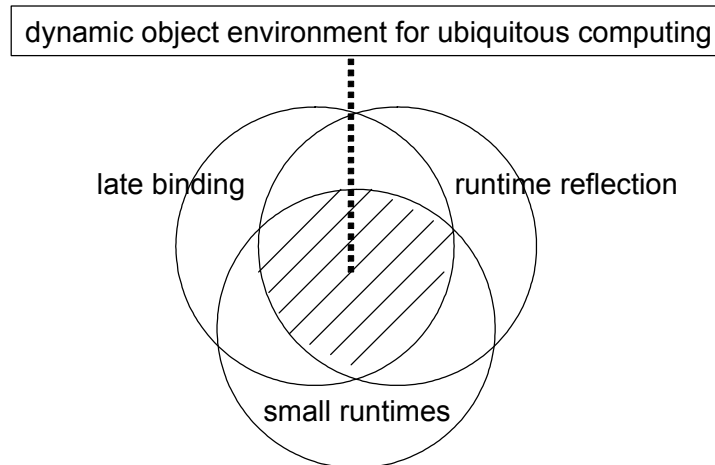


Figure 1: Dynamic object environment for ubiquitous computing

The availability of small reflective runtimes is essential for supporting flexible computing platforms especially on limited devices. Two of the candidates we consider promising are Resilient [5] and Squat [7]. Resilient is a Smalltalk-based software platform for embedded devices. A virtual machine, core libraries, device drivers, and a networking protocol stack are provided in less than 128 KB. Running code can be debugged and updated directly at the deployment platform, also. Squat is a portable Squeak [3, 7] that can start and extend itself. Its initial deployment size is less than 300 KB, consisting of an installer, a virtual machine, and a preliminary system snapshot to be expanded and shrunk over the network at runtime according to the desired system behavior at runtime.

So far we have been working on dynamic service adaptation [2] and context-dependent object behavior [1] which mainly exploits late binding and runtime reflection. We now investigate the extension of our work into the area of small runtimes to better support dynamic adaptation in pervasive computing environments.

References

- [1] R. Hirschfeld, and M. Wagner. PerspectiveS – AspectS with Context. In Proceedings of the OOPSLA 2002 Workshop on Engineering Context-Aware Object-Oriented Systems and Environments (ECOOSE), Seattle, WA, USA, November 5, 2002.
- [2] R. Hirschfeld, K. Kawamura, and H. Berndt. Dynamic Service Adaptation for Runtime System Extensions. In R. Battiti, R. Io Cigno, M. Conti, editors, Wireless On-Demand Network Systems, Proceedings of WONS2004, LNCS 2928, pp. 225-238, Springer 2004.
- [3] D. Ingalls, T. Kaehler, J. Maloney, S. Wallace, and A. Kay. Back to the Future: The Story of Squeak, a Practical Smalltalk Written in Itself. In Proceedings of the 1997 Conference on Object-Oriented Programming Systems, Languages and Applications (OOPSLA), pp. 318-326, Atlanta, GA, USA, October 1997.
- [4] P. Maes. Concepts and Experiments in Computational Reflection. In: Proceedings of the 1987 Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA), pp. 147-155, Orlando, FL, USA, 1987.
- [5] OOVm homepage (<http://www.oovm.com/>).
- [6] Squat homepage (<http://www.netjam.org/squat/>).
- [7] Squeak homepage (<http://www.squeak.org/>).