Rascal – A Resource Manager for Intelligent Environments

Krzysztof Gajos, Luke Weisman, Stephen Peters & Howard Shrobe

Artificial Intelligence Laboratory
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139

http://www.ai.mit.edu

The Problem: The problem is that of assigning abstract services provided by various devices (physical and computational) to requestors. Each device can provide a number of services. Each kind of abstract service can potentially be provided by a number of different devices with some variation of quality. The problem is to find the right devices for the requests while keeping conflicts to a minimum.

Motivation: Smart space technology is becoming increasingly more robust and accessible. But still one of the major obstacles preventing this technology from spreading is the fact that different smart spaces are equipped with very different kinds of devices. This lack of consistency makes it very difficult to write portable applications for intelligent spaces. It is thus necessary to provide an extra layer of abstraction to enable the creation of device-independent applications.

Furthermore, as the work on smart spaces progresses, the number of applications that can run independently in such environments is steadily increasing. When several applications run concurrently in an environment, a problem is created when they inevitably vie for scarce shared resources.

Previous Work: There are several other systems that deal with Resource Management issues in smart environments. Examples of such systems include Jini[1], Open Agent Architecture (OAA)[5], Hive[6], or Resource Description Framework (RDF)[4]. None of them, however, address all of the issues identified by us as crucial in designing a resource management system for intelligent spaces. Jini is only a communication, description and discovery framework with no management capabilities of its own. OAA manages tasks and not services. Hive does not have any explicit mechanism for automatic conflict avoidance and RDF only provides a description framework without any explicit support for reasoning.

Approach: Rascal [2] performs two fundamental tasks: *resource mapping* and *arbitration*. Resource mapping (i.e. match-making) is the process of deciding what resources can be used to satisfy a specific request. Arbitration is ensuring that, at a minimum, resources are not being used beyond their capacities. Ideally, arbitration ensures optimal, or nearly optimal, use of scarce resources via appropriate allocation of resources to requests.

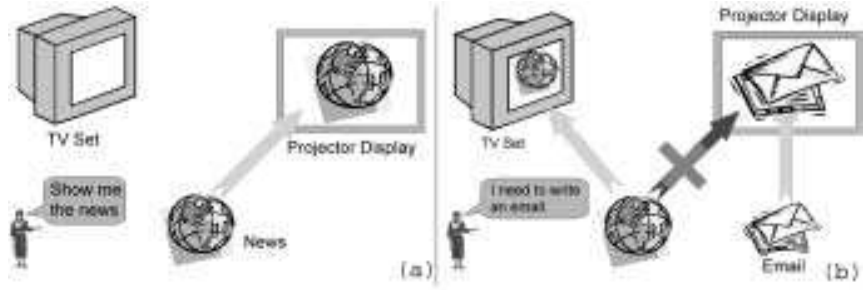

Figure 1: Sample interaction. (a) user requests to see the news – on-wall projected display is allocated as the best resource for the task. (b) user accesses email; the only possible display for email is the on-wall projector previously allocated to the news agent. Instead of being stopped, the news agent is moved to a TV set.

One of the features that sets Rascal apart from other resource management systems is that it is capable of performing "smart re-allocations" as shown in Figure 1 and explained in [3].

Impact: Deployment of a resource management infrastructure has fundamentally changed our approach to constructing applications for the Intelligent Room. They are now written in a device-independant fashion and can easily adapt to available resources in a variety of Metaglue-enabled spaces. The introduction of resource management into Metaglue made it possible to deploy our technology in nearly a dozen different offices throughout the lab.

Our work on Rascal has also led to identifying certain fundamental principles for building high-level resource management systems for smart spaces. Those principles are discussed in [3].

Future Work: Currently, Rascal manages resources of a single entitiy such as a person or a space. We are currently working on an extension of our resource management system that would make it possible for various entities to make requests of each other.

Research Support: This material is based upon work supported by the MIT Project Oxygen and the Advanced Research Projects Agency of the Department of Defense under contract number DAAD17-99-C-0060, monitored through Rome Laboratory. Additional support was provided by Microsoft Corporation.

References:

[1] Ken Arnold, Bryan O'Sullivan, Robert W. Scheifler, Jim Waldo, and Ann Wollrath. *The Jini Specification*. Addison-Wesley, Reading, MA, 1999.

[2] Krzysztof Gajos. Rascal - a resource manager for multi agent systems in smart spaces. In *Proceedings of CEEMAS 2001*. Springer, 2001. To appear.

[3] Krzysztof Gajos, Luke Weisman, and Howard Shrobe. Design principles for resource management systems for intelligent spaces. In *Proceedings of The Second International Workshop on Self-Adaptive Software*, Budapest, Hungary, 2001. To appear.

[4] Ora Lassila and Ralph R. Swick. Resource description framework (rdf) model and syntax specification. w3c recommendation. Technical report, World Wide Web Consortium, 1999.

[5] David L. Martin, Adam J. Cheyer, and Douglas B. Moran. The Open Agent Architecture: A framework for building distributed software systems. *Applied Artificial Intelligence*, 13(1-2):91–128, January-March 1999.

[6] Nelson Minar, Matthew Gray, Oliver Roup, Raffi Krikorian, and Pattie Maes. Hive: Distributed agents for networking things. In *Proceedings of ASA/MA'99, the First International Symposium on Agent Systems and Applications and Third International Symposium on Mobile Agents*, August 1999.