Mobile UNITY Coordination Constructs
Applied to Packet Forwarding for Mobile Hosts *

Peter J. McCann and Gruia-Catalin Roman
Department of Computer Science, Washington University, One Brookings Drive, St. Louis MO 63130

Abstract. With recent advances in wireless communication technology, mobile computing is an increasingly important area of research. A mobile system is one where independently executing components may migrate through some space during the course of the computation, and where the pattern of connectivity among the components changes as they move in and out of proximity. Mobile UNITY is a language and logic for specifying and reasoning about mobile systems, the components of which must operate in a highly decoupled way. In this paper it is argued that Mobile UNITY contributes to the modular development of system specifications precisely because of the decoupled and declarative fashion in which coordination among components is specified. The packet forwarding mechanism which is at the core of the Mobile IP protocol for routing to mobile hosts is taken as an example. A Mobile UNITY specification of packet forwarding and the mobile system in which it must operate is developed. Mobile hosts are the components that can disconnect from one location in the network and reconnect to another at any point during system execution. Finally, the role of formal program verification in the development of protocols like Mobile IP is discussed.

1 Introduction

Mobile computing represents a major point of departure from the traditional distributed computing paradigm. The potentially very large number of independent program units, a decoupled computing style, frequent disconnections, continuous position changes, and the location-dependent nature of the behavior and communication patterns present designers with unprecedented challenges in the areas of modularity and dependability. So far, the literature on mobile computing is dominated by concerns having to do with the development of protocols and services for this environment. These services are characterized by more dynamic binding and weaker consistency than traditional distributed applications.

* This paper is based upon work supported in part by the National Science Foundation under Grant No. CCR-9217751. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the National Science Foundation. The authors thank Toni Reiss for the illustrations in figures 1 and 3.
For example, the components needed to carry out a service are often not determined until runtime, as in the location-dependent services provided by a mobile web browser [12]. Other work has pointed out the importance of context other than location [9], such as the presence or absence of other components. Weak consistency protocols for filesystems and databases [8, 10, 11] are motivated by the low bandwidth and frequent disconnections typical of a wireless network with mobile nodes. These systems trade consistency for availability under the assumption that in some cases, dealing with the consequences of inconsistencies is cheaper than denying access to a resource.

Some researchers have focused on toolkits and abstractions for building mobile applications. Badrinath and Welling [1] describe a C++ abstraction for delivering events such as bandwidth variations, disconnections, and battery measurements to applications. Noble, Price, and Satyanarayanan [6] present the *Odyssey* application library for managing changing resources and emphasize the importance of application- and data type-specific policies for reacting to changes in the environment. Both emphasize the need to present information about connectivity directly to applications, which violates traditional notions of abstractions and encapsulation of the network. Such information is necessary, however, to build applications that behave properly under changing circumstances, such as responding to diminished connectivity by changing to a lower-resolution video stream.

In this paper we focus on new kinds of abstractions for interprocess communication in the mobile setting. Mobile UNITY [5] provides a notation for mobile system components and a coordination language for expressing interactions among the components. Once expressed in our notation, a system can be subjected to rigorous formal verification against a set of requirements expressed as temporal properties of executions. Mobile UNITY is based on the UNITY model of Chandy and Misra [2], with extensions to both the notation and logic to accommodate specification of and reasoning about mobile programs. In Mobile UNITY, each program is a unit of mobility and all variables are locally owned. We capture movement by augmenting the program state with a location attribute whose change in value is used to represent motion. The new language supports a declarative style of communication that allows a component program to be written in a modular fashion, without regard to the identities of the other components with which it must later interact. This is accomplished with a novel construct, transient variable sharing. This allows mobile programs to share data when in close proximity, i.e., a variable owned by one program may be shared in a transparent manner with different programs at different times depending upon their relative location in space. This implies that a value written to one variable of such a pair must be propagated to the other variable as a side-effect of an assignment. The basic constructs of Mobile UNITY that allow us to express this idea also allow for other coordination constructs, such as transient statement synchronization. However in this paper we deal only with transient sharing, which suffices for the examples presented.