Workshop on Software Engineering and Mobility

Gruia-Catalin Roman
Dept. of Computer Science
Washington University
One Brookings Drive, St. Louis, MO 63130, USA
roman@cs.wustl.edu

Gian Pietro Picco
Dip. di Elettronica e Informazione
Politecnico di Milano
P.za L. da Vinci 32, I-20133 Milano, Italy
picco@elet.polimi.it

Abstract

Mobility is redefining the hardware and software fabric of distributed systems. Wireless communication allows network hosts to participate in a distributed computation while on the move. Novel middleware technologies allow software components to migrate across hosts for enhanced flexibility or performance. Workshop participants were invited to analyze the software engineering implications of this wave of technological changes, by discussing fundamental models, emerging themes, research opportunities, technological trends, and market forces.

The last decade of the twentieth century was characterized by a massive integration of computing and communications. So much so that a network connection temporarily being down is often equated with rendering the computer as useless as an unplugged telephone, despite the local services the CPU can continue to provide. This happens, not because the computer operation is necessarily dependent on network resources, but due to the fact that our every day activities rely on a tight and responsive integration of computing and communication capabilities. The next step in this consolidation trend is likely to entail the socio-economic assimilation of the computing and communication infrastructure. Pervasive computing is a term often used to refer to the outcome of this process. This capacity to access information at any time and in any place is made possible by the widespread availability of the Internet, by the embedding of miniaturized computing/communication capabilities in everyday devices, and by an explosion in wireless communication. Recent technical developments and powerful social trends are starting to challenge our ability to support pervasive computing and communication under all circumstances. A revolution in sensor technology is setting the stage for the deployment of systems involving hundreds and even thousands of semi-autonomous components. The movement of people, vehicles, and devices forces us to consider systems whose structure is spatially distributed and subject to continuous evolution.

In the traditional static network, fixed hosts with statically assigned addresses exchange messages via the standard Internet infrastructure of fixed routers and switches. At the periphery of this fixed network one can envision base stations (fixed routers with wireless communication capabilities) that control message traffic to and from mobile hosts forming a dynamic fringe. Ultimately, mobile hosts can detach themselves completely from the fixed infrastructure and may evolve independently of it. Such clouds, called mobile ad hoc networks [2], are opportunistically formed structures that change rapidly in response to the movement of the carriers to which the mobile hosts are attached. For the time being, communication within ad hoc networks is point to point over a physical broadcast medium, the airwaves. However, routing in ad hoc networks will expand the connectivity pattern beyond the limits of an immediately accessible region.

Across this highly dynamic physical structure a logical layer still more fluid is emerging. Code mobility [1] removes the static binding between the software components of a distributed application and the network hosts where they are executing. Relocation of such components is then enabled, to achieve a higher degree of flexibility and customizability, or improved bandwidth utilization. More radical views encompass migration of execution units, mobile agents autonomously performing tasks without requiring permanent connectivity towards the client. Although the interest in code mobility is being popularized and amplified by the success of the Java language, the relevance of mobile code is not only technological. This form of logical mobility has the potential to completely change the way distributed applications are conceived and deployed.

Mobility [3] is an area of growing importance, affecting all sectors of life and of economy. It can enhance our quality of life, our productivity, our ability to respond to emergencies, and our capacity to defend against military threats. Two features capture the essence of this application domain: it is open and dynamic. Even in the most controlled of cir-

cumstances as might be the case in a covert military operation, the requirement to respond to unexpected losses or new threats demands the ability to increase the level of participation of entities not initially included in the mission. Change is triggered by the physical mobility of hosts that affects the underlying network configuration and resource availability, by the logical mobility associated with the application code, and by the evolving needs of the end user. Successful development of mobile applications requires a software infrastructure that fits the character of the application and reduces the complexity application developers face. No such infrastructure exists today, and its development entails both significant new intellectual developments and a delivery mechanism that can be readily adopted in an industrial setting.

Sorting out the software engineering implications of this intriguing and perplexing wave of technological changes is a nontrivial task. This Workshop is dedicated to the notion that software engineering must play a constructive role during these early and exciting phases in the development of the infrastructure and applications supporting ubiquitous and mobile computing. Researchers were invited to discuss fundamental models, emerging themes, research opportunities, technological trends, and market forces in the field of mobile computing and communication. The immediate objective is to provide a forum for intellectual debate. The ultimate goal is to define an influential research agenda for the area as a whole and to generate advocacy for it by stimulating new research initiatives.

A central element of such an agenda must be the integration of research themes at all levels including models, technology, and applications. The resulting new perspective on mobility must be reflected in the models we use for analysis and in the software infrastructure that is the basis for application development. The most obvious dichotomy that must be reevaluated is the sharp delineation among the research communities involved in work with physical and logical mobility. We generally view physical mobility as a feature of the computing and communication environment and logical mobility as a design paradigm. While this distinction will not disappear, there are important opportunities for bridging the two cultures. First, the kind of applications that are enabled by physical mobility can achieve the desired level of flexibility and dynamic reconfigurability more readily by exploiting the design options made available by code mobility. A simple application such as renting a car may ultimately entail connectivity to wired networks to verify the identity of the driver, access to base stations to gather local weather data, and interactions in an ad hoc setting to coordinate traffic flow with other cars traveling on the same highway. Similarly, code specific to the rental contract may be downloaded on the vehicle upon exiting the parking lot, cars having the same origin of manufacture may spread updates of the latest tire pressure monitoring protocol as the pass each other, and entire applications may drift over the wave of vehicles to monitor traffic patterns. Second, support for smooth transition among different kinds of networks, from wired to ad hoc, by necessity requires the integration of the notions of movement in physical and logical spaces. Many applications must be able to execute in both settings (and any forms in between) and must adapt readily to changes in the underlying network structure. To accomplish this, our view of mobility must allow us to both ignore and differentiate among different forms of mobility as the need arises. Finally, a different kind of integration is required when considering the tradeoffs involved across layers in a mobile system. Communication protocols, middleware, and application needs can interact with each other in complex ways along multiple dimensions including dependability, functional capabilities, and performance. Guaranteeing data consistency in an ad hoc setting plagued by frequent unannounced disconnections may be impossible to achieve, in general. Yet, reasonable approximations can be built if one takes into consideration the application profile, e.g., pattern of interaction, location dependencies, or relative velocities.

By assembling together a diverse set of people with expertise in a broad range of topics relating to mobility, e.g.,

- applications, case studies,
- computational models, specification, verification,
- language constructs, algorithms, design patterns,
- software architectures, middleware,
- runtime systems, computing environments,
- development tools,
- communication protocols,

we hope to create the milieu necessary to focus our attention on themes that integrate rather than differentiate our perspectives on mobility. The Workshop is organized around a critical evaluation of submitted position papers with ample time for thoughtful discussion and debate. Invitations to present selected position papers and lead the related discussions were extended to authors whose work and ideas promise to be influential, are thought provoking, or offer an interesting integration of existing research trends.

References

[1] A. Fuggetta, G. P. Picco, and G. Vigna. Understanding Code Mobility. *IEEE Transactions on Software Engineering*, 24(5):342–361, May 1998.

[2] M.Corson, J.Macker, and G.Cinciarone. Internet-Based Mobile Ad Hoc Networking. *Internet Computing*, 3(4), 1999.

[3] G.-C. Roman, G. P. Picco, and A. L. Murphy. Software Engineering for Mobility: A Roadmap. In A. Finkelstein, editor, *The Future of Software Engineering*, pages 241–258. ACM Press, 2000. Invited contribution.