Adaptive Mobile Applications

Thomas Kunz
Carleton University, Canada

Abdulbaset Gaddah
Carleton University, Canada

INTRODUCTION

The convergence of two technological developments has made mobile computing a reality. In the last few years, developed countries spent large amounts of money to install and deploy wireless communication facilities. Originally aimed at telephone services (which still account for the majority of usage), the same infrastructure is increasingly used to transfer data. In parallel, wireless LAN technologies are providing hotspot coverage in many high-traffic locations. The second development is the continuing reduction in size of computer hardware, leading to portable computation devices such as laptops, palmtops, or functionally enhanced cell phones. Given current technology, a user can run a set of applications on a portable device and communicate over a variety of communication links, depending on his/her current location.

As will be explained in more detail later on, the mobile computing environment is highly dynamic. Available bandwidth changes by orders of magnitudes, based on the selected wireless access technology. Also, portable devices differ in processing power, memory, display capabilities, and other characteristics. It is generally argued that applications should "adapt" to the current environment, for example by filtering and compressing data or by changing the functionality offered to the user. Some researchers even argue that all future applications, not just the ones intended for execution on mobile devices, will have to be able to adapt to changing requirements and changing implementation environments on time scales from microseconds to years (Kavi, 1999). This article reviews the work on adaptive mobile applications and provides an outlook on future trends.

The alternative to adaptive applications is to either implement a single application that is designed for the lowest common denominator (in terms of resource availability) or multiple functionally identical or similar binaries, tuned for specific environments. The former will needlessly sacrifice application features when running in more resource-rich environments. The latter approach is an inferior solution as well, for a number of reasons. The user of a portable device has to install and maintain multiple applications, which is a drain on the limited storage capabilities typically found on those devices. It also potentially results in different user interfaces and causes high software development overheads when developing the "same" mobile application multiple times. Finally, it forces the user to identify the current execution conditions and select the "right" application.

The next section will review the motivation for adaptive approaches towards mobile application design. We will then briefly review traditional approaches to adaptive mobile applications, followed by a discussion of mobile middleware that is intended to support adaptive mobile applications. The article finishes with a brief conclusion of the state-of-the-art and identifies areas of future work.

BACKGROUND

Wireless communication and portable devices make it possible for mobile users to have access to information anywhere and anytime. Designing, implementing and deploying applications that work well across all portable devices and across a wide range of wireless access networks is non-trivial.

There are at least three common factors that affect the design of mobile applications: portable devices, network connection, and mobility. *Portable devices* vary from one to another in term of resource availability. Devices like laptops can offer fast CPUs and large amount of RAM and disk space while others like pocket PCs and phones usually have scarce resources. It is either impossible or too expensive to augment the resource availability. Hence, applications should be designed to achieve optimal resource utilization. In general, the design of portable devices strives for properties such as size, weight, durability and long battery life. Different devices will emphasize different trade-offs between CPU speed, memory, I/O capabilities, and power consumption, providing very heterogeneous execution environments for mobile applications.

Network connection in mobile scenarios is characterized by limited bandwidth, high error rate, higher cost, and frequent disconnections due to power limitations, available spectrum, and mobility. Wireless communication is more difficult to achieve than wired communication because the surrounding environment interacts with the signal, blocking signal paths and introducing noise and echoes. Therefore, mobile application designs need to be more concerned about the network conditions than applications designed for fixed networks. Many wireless and mobile networks such as WaveLAN are organized into geographically defined cells, with a control point called a base station in each of the cells. Devices within the same cell share the network bandwidth; hence, the bandwidth rapidly decreases whenever a new device joins the cell. Portable devices may move around different areas with no coverage or high interference that cause a sudden drop in network bandwidth or a loss of connection entirely. Unpredictable disconnection is also a common issue that frequently occurs due to the handoff process or shadowed areas. Most wireless network services charge a flat fee for their service, which usually covers a fixed number of messages. Additional charges are levied on per packet or per message basis. In contrast, the cost for sending data over cellular networks is based on connection time instead. This forces mobile users to connect for short periods of time.

Physical device mobility can greatly affect network connection, which accordingly has to adapt to user mobility by reconnecting the user with respect to a new location. Portable devices may interact with different types of networks, services, and security policies as they move from one area to another. This requires applications to behave differently to cope with dynamic changes of the environment parameters. As a consequence, mobile applications also have to cope with a much greater variation in network bandwidth: bandwidth can shift from one to six orders of magnitude between being plugged into the wired network versus using (different) wireless access networks.

The constraints and limitations mobile applications face are not a product of current technology, but they are related naturally to mobility. Together, they complicate the design of mobile applications and require rethinking traditional approaches to application design. Any feasible approach to mobile computing must strike a balance between conflicting demands. This balance cannot be static, as the execution environment of mobile applications varies; it must react, or in other words, the applications must be adaptive.

ADAPTIVE MOBILE APPLICATIONS: TRADITIONAL APPROACHES

Designing adaptive applications is an active research area. Traditionally, most work focused on the wireless link(s). Early work provides general solutions that do not change the TCP semantics but focus on improving TCP throughput over wireless links; see for example Balakrishnan (1995). While this addresses issues such as high link error rates and spurious disconnections, it does not address the low and highly variable bandwidth characteristic of mobile computing.

A second group of approaches adapts to the scarce and varying wireless link bandwidth by filtering and compressing the data stream between a client application on a portable device and a server executing on a stationary host. Data compression is done at one of two places. Bolliger (1998) and Seshan (1997) enhance the server to generate a data stream that is suited for the currently available bandwidth. This typically represents an end-to-end approach, which is well known in the networking and system literature. Most other proposals (Angin, 1998; Fox, 1998) extend the client-server structure to a client-proxy-server structure, where a proxy executes in the wireless access network, close to the portable device. This proxy-based approach filters and compresses the data stream originating from the server to suit the current wireless bandwidth. Joshi (1997) incorporates both end-to-end and proxy-based approaches, using each as appropriate, to support Web access from mobile platforms. For example, tasks such as complex filtration, which require significant computational resources, are done in an end-to-end manner. The proxy-based approach, on the other hand, is used when the server is not able to generate the appropriate data stream.

A third, complementary approach, focuses on the computational effort (Kunz, 2000). Mobile applications, especially ones that require intensive computation (for example, video decoding), can be divided dynamically between the wired network and the portable device according to the mobile environment and to the availability of the resources on the portable device, the wireless link, and the wired network. The access network supports the mobile application by providing proxy servers that can execute parts of the application code. This may increase the performance of applications and reduce the power consumption on portable devices since offloading computation to the proxies in the wired network will reduce the CPU cycles and memory needed to achieve certain tasks at portable devices.

FUTURE TRENDS: MOBILE MIDDLEWARE

The early approaches reviewed in the previous section typically provide toolkits that support specific adaptation ideas. To generalize this effort, support for adaptive mobile applications should be embedded into appropriate mobile middleware. Traditional middleware systems, like CORBA and DCOM, have achieved great success in dealing with the heterogeneity in the underlying hardware and software platforms, offering portability, and facilitating development of distributed applications. However, these systems are based on the assumptions that the distributed applications will run in a static environment; hence, they fail to provide the appropriate support for mobile applications. Therefore, mobile applications need a middleware that facilitates adapting to environment variations.