Mobile Application Profiling for Connected Mobile Devices

The SymPA (Symbian Protocol Analyzer) tool correlates traffic information, radio-access-technology measurements, and location data to help developers evaluate mobile applications in the field.

Almudena Díaz, Pedro Merino, and F. Javier Rivas
University of Málaga

The recent appearance of open operating systems and virtual machines optimized for mobile devices has led to new and innovative applications.[1] However, although these mobile devices pave the way for a new generation of applications that exploit their intrinsic mobility, proximity to the user, and dependence on handheld devices, they create unique challenges for developers.[2] For instance, new mobile devices come with powerful multitasking processors, but their performance and processing capacity differ significantly from those of traditional desktop computers. In this restricted and unknown development environment, developers face new constraints related to not only hardware issues such as processors and memory but also the lack of tools for debugging the different aspects that affect mobile application performance.

Mobile device technologies such as Bluetooth, wireless local area networks (WLANs), General Packet Radio Service (GPRS) or Enhanced Data Rate for Global Evolution (EDGE), Universal Mobile Telecommunications System (UMTS)/High-Speed Packet Data Access (HSDPA), and WiMAX also introduce variables during development that developers working on fixed networks are unaccustomed to. Each technology has different behaviors and is suitable for different contexts. In addition, the simultaneous use of different radio-access technologies is a new challenge for mobile developers.

Several existing tools can analyze a mobile application's resource consumption performance. However, these tools don't address communication performance from a developer's standpoint (see the "Related Work in Application Profiling" sidebar) or answer these questions:

- How can I debug the application's data connection behavior?
- What's my protocol's performance?
- Can I improve my application's connection performance?
- What's my application's connection quality?
- What's my real available bandwidth?

Such questions proliferate in dynamic scenarios in which propagation conditions are less stable and vertical handover occurs between different radio-access technologies.

To provide the answers to these and other questions, and to help developers find the cause of errors related to accessing network resources, we created SymPA (Symbian Protocol Analyzer; www.lcc.uma.es/~pedro/mobile). SymPA is a free software tool with packet-sniffing features, measurement localization maps, and other relevant functionalities (see

Published by the IEEE CS ■ 1536-1268/10/$26.00 © 2010 IEEE

Related Work in Application Profiling

Symbian wireless network performance analysis tools, such as QualiPoc (www.swissqual.com), Nemo Handy (www.anite.com), and Customer Experience Manager (www.mformation.com), provide low-level information and focus on specific service tests. However, unlike SymPA, they don't let developers debug their protocols' behavior, solve connectivity problems, or correlate these issues with location data.

Other solutions focus on providing on-target tests and debugging capabilities. JInjector is an instrumentation tool that lets developers evaluate code coverage for Java 2 Micro Edition applications both in emulators and on the actual devices.[1] The ability to debug applications on real devices is indispensable because applications usually work successfully on the emulator but fail on real targets. On-target testing is also one of the most time-consuming tasks and therefore requires simplification and facilitation.

Forum Nokia's Remote Device Access (RDA) service offers remote access to a wide range of mobile devices, so developers can test their applications on a pool of real devices. RDA also provides the possibility of testing on prototypes, which lets developers design new applications that exploit these new devices' functionalities. Similar initiatives from Forum Nokia are the Virtual Developer Lab, powered by DeviceAnywhere, and the Device Loaner Program. These initiatives reveal the importance of using real devices during mobile application development.

Deploying the application on real devices during testing is also necessary to analyze application performance on the hardware for which it was designed. The Nokia Energy Profiler lets developers monitor their applications' energy use in real time and analyzes the processor, memory, and network signal levels. The performance investigator delivered with Carbide.c++, an Eclipse-based development tool supporting Symbian OS development, enables on-target data tracing of function calls, power use, memory use, and key events. Mobile application evaluation requires not only using real devices but also testing them in the field.[2] SymPA addresses this key issue of mobile application development.

Finally, stability and security are crucial. Stability is key for mobile devices because they must operate for long time periods. Users must always be able to switch away from any executing application and make an emergency call. Symbian Signed (www.symbiansigned.com) provides a unified testing and certification process to ensure that devices operate normally after applications are installed or removed and that applications are safe for users and don't harm network operation. This initiative, as well as Java Verified (www.javaverified.com) and True Brew Certification (http://brew.qualcomm.com), reveals that application integrity concerns not only end users but also developers, mobile network operators, and phone manufacturers.

The mobile sector's different stakeholders are clearly aware that the correct performance of mobile applications is vital for the technological evolution of devices, applications, and services. But it's also clear that communications performance, an open issue in application development,[3] is beyond the scope of current development support tools.

REFERENCES

1. M. Sama and J. Harty, "Using Code Instrumentation to Enhance Testing on J2ME: A Lesson Learned with JInjector," *Proc. 10th Workshop Mobile Computing Systems and Applications*, ACM Press, 2009, pp. 1–6.
2. T. Halonen, J. Melero, and J.R. Garcia, *GSM, GPRS and EDGE Performance: Evolution toward 3G/UMTS*, Halsted Press, 2002.
3. K.A. Brown, "Impact of Wireless Communication on Multimedia Application Performance," *Multimedia Systems and Applications* (*Proc. SPIE*, vol. 3528), 1999, pp. 566–573.

Figure 1) that let developers profile mobile applications with advanced communication capacities. We developed SymPA using Symbian, the most extended OS for mobile phones, according to the Canalys report "Smart Mobile Device Shipments Hit 118 Million in 2007, up 53% on 2006" (www.canalys.com/pr/2008/r2008021.htm). Symbian supports an ample range of programming languages, including Symbian C++, native C, native C++, Java, Python, Adobe Flash Lite, and Ruby, and lets developers test applications programmed in different languages. University of Málaga researchers have successfully tested SymPA on projects ranging from mobile application development to mobile service characterization over cellular networks.[3,4]

New Profiling Needs in "Always Connected" Applications

Because programmers need to manage new APIs, OSs, and implementation methodologies, mobile device software development has a long timescale.[5] Networking is one of the most important issues developers deal with during implementation and testing on real devices. Guaranteeing mobile application networking performance is key for the future of mobile devices. One reason this new generation of mobile phones has been designated as "smart" is their capacity to communicate with many smart networks, ranging from near-field communication to

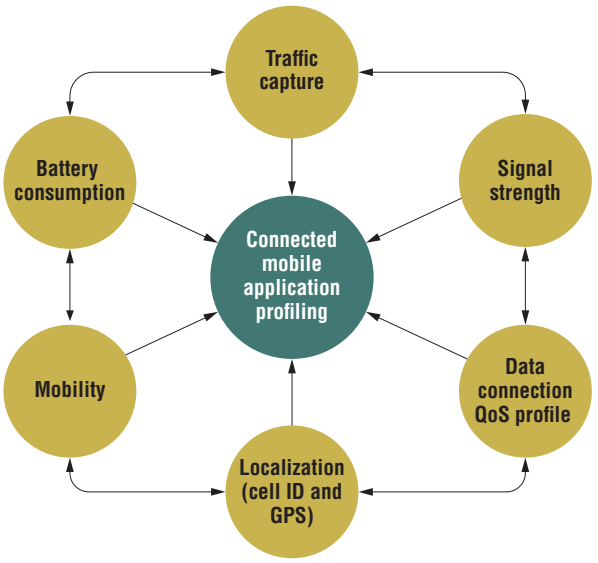


Figure 1. Mobile application profiling. A complete characterization of mobile applications requires monitoring and correlating different types of information, such as IP traffic, received signal strength, location, and battery consumption.

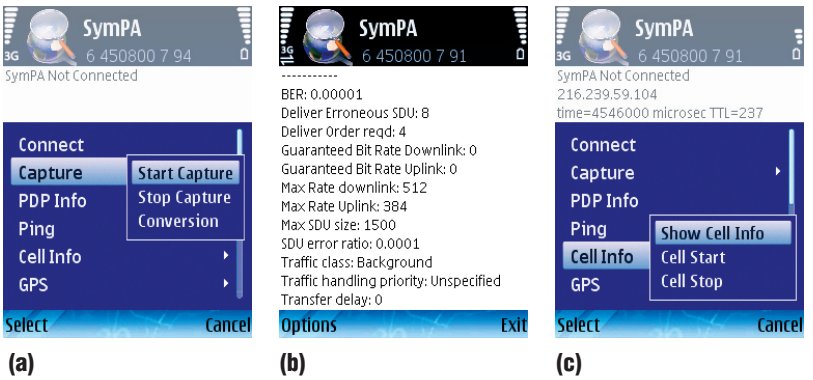


Figure 2. SymPA (Symbian Protocol Analyzer) running on a Nokia 6110 Navigator. The mobile device provides advanced monitoring functionalities directly: (a) IP traffic capture, (b) quality-of-service (QoS) profile parameters, and (c) cell information.

global communication using cellular technologies.

Wireless network and mobile phone networking issues require more effort than desktop computer issues because developers lack tools to check connectivity, analyze traffic data, study protocol stack configuration, and debug mobile communication performance errors. For example, implementing servers on mobile devices is tedious because of limitations imposed by network operators. In this situation, developers should be able to determine whether a problem is related to closed ports or server implementation. This issue might seem trivial, but developers must analyze more complex issues related to mobile communications' changeable nature to discern between application performance issues or network errors.

Developers should also be able to cope with issues such as link outages. However, to implement such functionality, they need tools to analyze traffic and correlate it with location, speed, and wireless network issues. Improving the application's radio technology battery consumption is also a concern.

The protocol stack configurations on mobile devices differ from those of traditional PCs, so developers can't use emulators. We don't recommend using a mobile phone as a modem because applications tested this way use the computer protocol stack to establish data connections. So, developers can't use traditional protocol analyzers to study networking performance. To analyze connectivity issues, developers must execute the applications on real devices, which requires tools that run on these devices. In addition, adequate protocol behavior debugging tools might reduce development and testing time, but existing on-device debugging tools can detect only coding errors, not communication underperformance. The SymPA profiling tool runs on real devices to help developers deal with these issues (see Figure 2).

SymPA also supports heterogeneous environments. Heterogeneous networks are one of the most interesting possibilities mobile devices offer, promising the development of "always connected" applications. However, heterogeneous environments are critical scenarios in which connectivity could become a challenge for mobile developers. SymPA enables performance evaluation of vertical handover on mobile devices, which should allow seamless roaming between different networks. Developers could use SymPA to analyze these techniques' impact on IP communications from

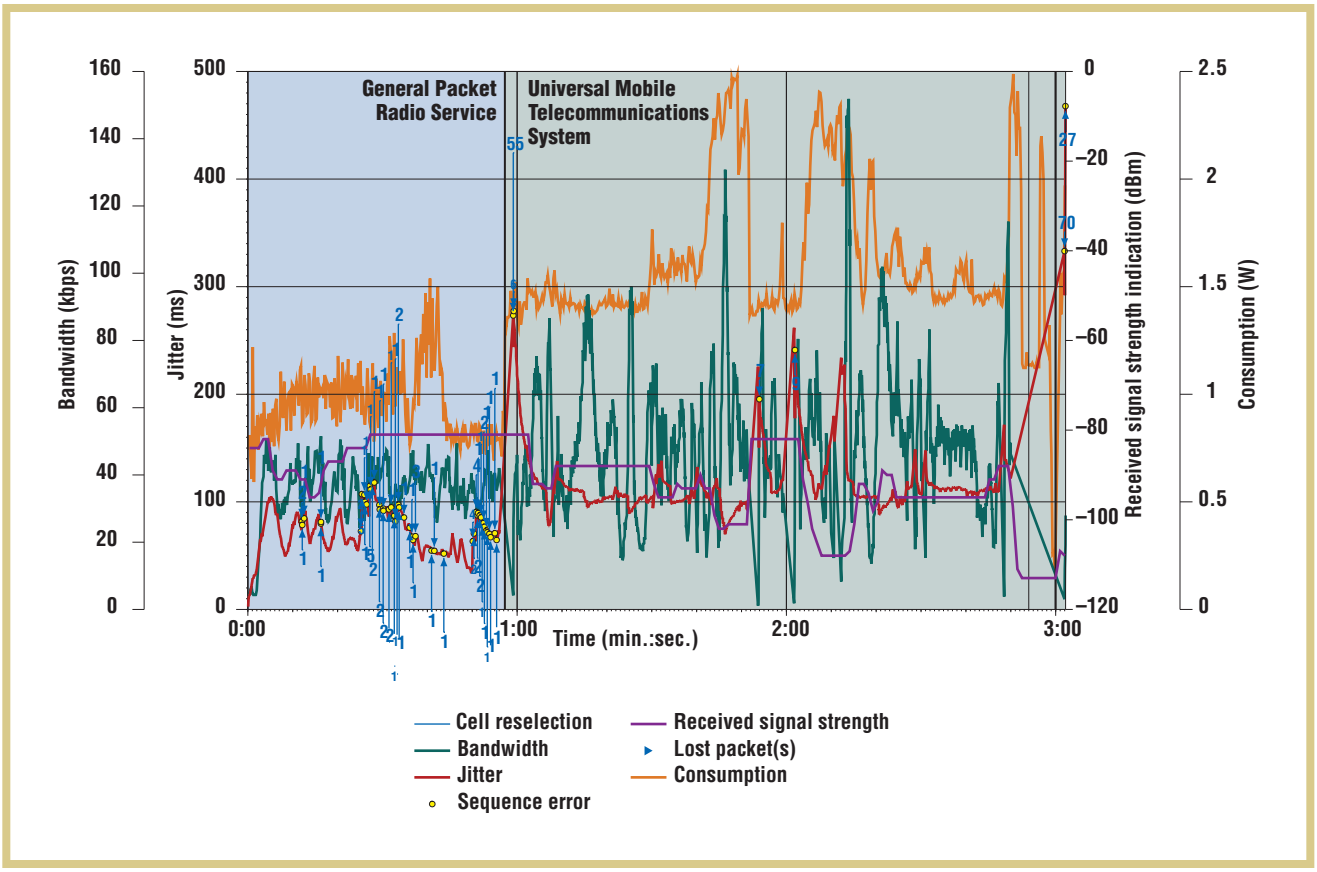


Figure 3. Handover between General Packet Radio Service (GPRS) and Universal Mobile Telecommunications System (UMTS) networks. The handover causes a burst of packet losses, jitter increase, and bandwidth decrease. After handover, UMTS provides higher bandwidth, reducing the amount of lost packets at the expense of power consumption.

the application's viewpoint. For example, we've used SymPA to characterize handover between GPRS and UMTS networks. Figure 3 shows the burst of lost packets during a handover on a streaming session in a vehicle test. It also shows each technology's signal strength, jitter, bandwidth, and battery consumption.

Enabling Mobile Application Analysis

SymPA analysis centers on networking issues. In general, radio propagation conditions become unstable in wireless scenarios, where factors such as signal quality and received signal strength—usually affected by Doppler and multipath fading effects—trigger cell reselections and handovers between different access technologies. Analyzing this information is essential to identifying the source of connectivity issues such as call drops, packet losses, or bandwidth reduction. SymPA helps monitor all these parameters.

Figure 4 shows SymPA's architecture, including the libraries, APIs, and different modules we implemented to provide its analysis functionalities—analyzing data connections, checking connectivity, localizing measurements, tracking energy consumption, and correlating data.

Analyzing Data Connections

SymPA's data connection analysis functionality includes traffic capture, received-signal-strength tracking, and radio-access-technology identification. SymPA also reports the quality-of-service (QoS) profile granted by cellular networks during the establishment of data connections.

Data traffic capture is fundamental for not only traffic analysis but also debugging communication protocol implementations. SymPA users can apply a wide range of measurements to the established connections to obtain parameters such as bandwidth, delay, and jitter. Other useful utilities include incoming establishment-availability checking, which can determine whether a mobile operator has closed ports.

The mobile terminal's memory stores the raw data traffic, which can be exported to libpcap. Because libpcap is a widely used standard format, it's compatible with analysis tools such as Wireshark, a well-known protocol analyzer.

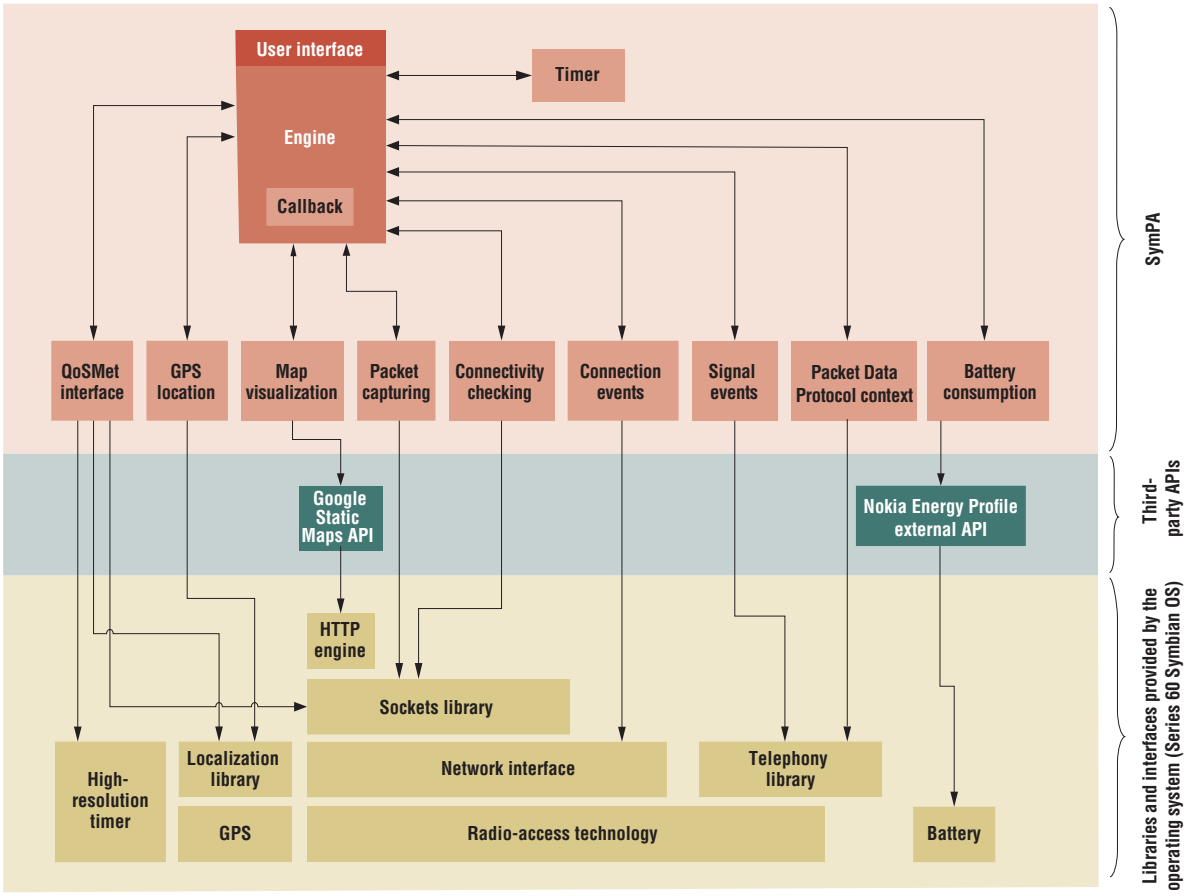


Figure 4. SymPA architecture splits functionalities into different modules. Some modules use third-party APIs, but the Symbian OS provides the main APIs.

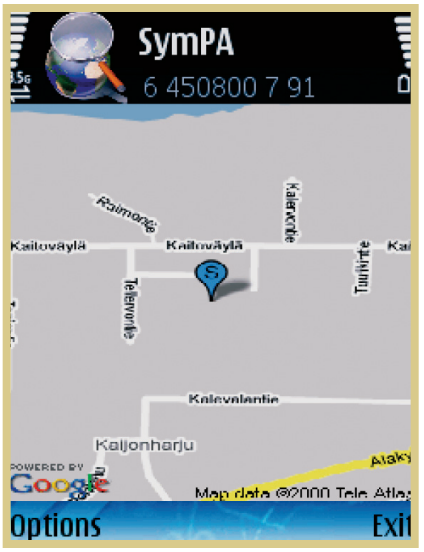


Figure 5. Geographic localization on a mobile device. SymPA provides positioning features to locate measurements using maps.

Checking Connectivity

SymPA lets developers verify mobile-to-Internet connectivity by establishing a data connection and then obtaining the IP address associated with the connection. Obtaining the IP address is useful because it changes each time a connection is established and can be used for further data processing or to accept connections in servers running in mobile devices.

Checking connectivity is particularly useful in mobile-to-mobile use cases because developers can't deploy traditional tools on the server side, as in a typical Internet scenario. SymPA incorporates a TCP client and a server, which allows file transfer between mobile devices. This active traffic generation utility, in conjunction with the traffic capture feature, enables mobile-to-mobile connection performance evaluation.

A standard ping functionality lets users obtain an estimate of the connections' round-trip delay. Developers need a proper characterization of communication delays to design timers

and buffers adapted to the actual conditions that mobile applications will experience.

Localizing Measurements

SymPA includes two features that accurately locate the measurements collected and determine the associated speed for each piece of information. First, the GPS-tracking function activates monitoring of the mobile terminal's speed and position. GPS-related information, such as time stamps and coordinates, is stored in a file. Developers can use time stamps during postprocessing to match different information sources such as captured traffic and radio condition information.

Second, the Google Static Maps API gives developers a map of the area directly on a mobile device (see Figure 5). With this information, identifying roads, buildings, and other objects that could affect communication is easy. In addition, through postprocessing of the SymPA-generated data files (see Figure 6), developers can create more meaningful and detailed maps, including coverage and handovers on a user's path.

Figure 7 shows how the GPS-tracking functionality's location information allows a deeper understanding of user mobility communication issues. To generate this representation, we used Google Earth, which integrates easily with SymPA. We tracked the received signal power of a user who was accessing an audio-streaming service while riding a bike from the VTT Technical Research Centre of Finland in Oulu to the city center. The blue vertical lines with numbered marks on top correspond to changes from one cell to another.

During the experiment, degradation in the sound quality and a temporary interruption occurred at some point near the river. After analyzing the captured information, we determined that the problems weren't related to the server side or network congestion, but rather to a significant loss of received signal strength at the boundary between two cells. In the figure, it's evident that the degradation occurs around the change to cell 450787 and that the signal strength returns to acceptable levels on the other side of the river.

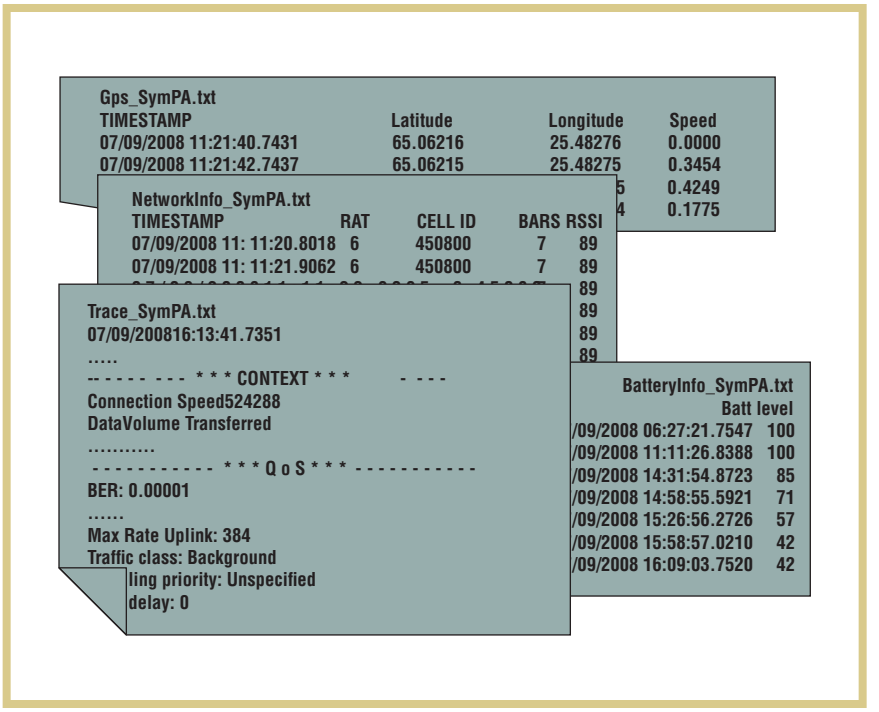


Figure 6. SymPA profiling files use a text-readable format to store measurements and timing information. Postprocessing time stamps enable correlation of different information sources, such as cell changes and packet losses.

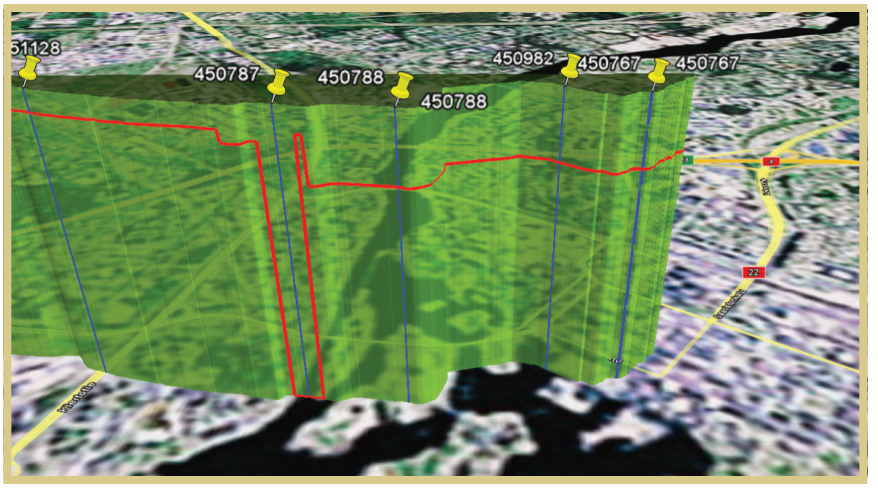


Figure 7. Correlation of Real-time Transport Protocol (RTP) traffic data, signal strength, and cell data. The red line represents the received signal power during a commute. The blue vertical lines correspond to changes from one cell to another. The signal degrades around the change to cell 450787 and returns to acceptable levels on the other side of the river.

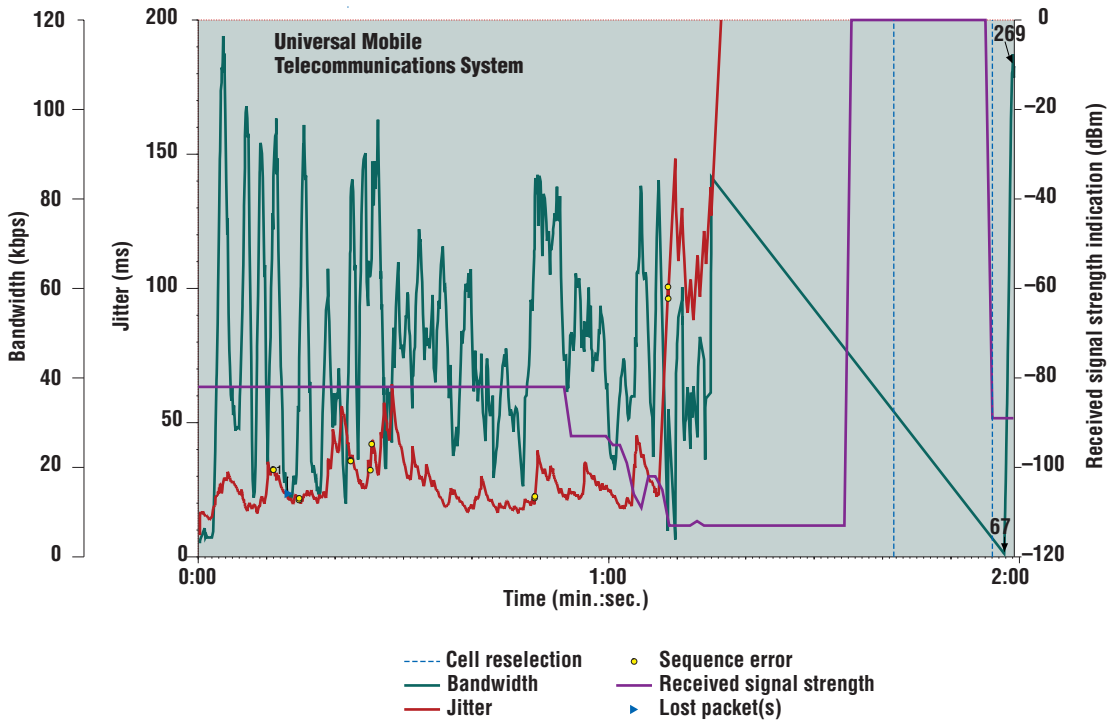


Figure 8. Data communication performance analysis on a mobile device. SymPA detects a link outage where the received signal falls for 20 seconds, interrupting the data connection. After 40 seconds, the communication continues in a new cell but loses 269 packets. In this way, SymPA helps identify the source of data connection errors.

Tracking Energy Consumption

Because mobile devices are battery driven, they require energy-saving strategies.[6] As a common practice, engineers need to pinpoint sources of higher energy consumption and concentrate later optimization on those critical functionalities. For that purpose, we added battery-monitoring features to SymPA.

In particular, SymPA lets developers fuse battery-consumption information with all the other available information so that they can associate results with the actual conditions during the measurement. Traditional profiling tools obtain the energy-related information regardless of the spatial information, which might negatively affect results.

For example, in a static environment, comparing only the power required in a WLAN connection to that required in a UMTS connection could lead to confusion. This is because the uplink power in UMTS might vary from insignificant values up to 2 watts, depending on the device's capabilities, the distance to the base station, and the propagation conditions.[7] Figure 3 shows how handover from GPRS to UMTS affects energy consumption during a streaming session. As expected, energy consumption is higher in UMTS, and the consumption increase is more significant when the received signal strength decreases. SymPA lets developers evaluate their applications' battery consumption using different radio-access technologies in each location and context.

Correlating Data

Correlating data from different levels lets developers find the source of errors at the application level and associate them with locations. For example, while profiling a video-streaming client over a public cellular network in a vehicular scenario, we detected a streaming session that finished abruptly. So, we used SymPA information to determine the problem's source.

Streaming quality depends on four main factors: bandwidth, packet loss, delay, and jitter. To calculate these factors, we analyzed Real-time Transport Protocol (RTP) traffic captured during a streaming session with Wireshark, then correlated this information with signal strength and cell data (see Figure 8). Signal strength degraded significantly below –110 dBm, a typical sensitivity threshold for mobile devices.

The signal dropped for 20 seconds. This link outage caused the video-streaming client's buffer to underflow, and the application stopped executing.

Adding location information to this correlation process, we determined that this error occurred when the vehicle was in a tunnel.

In the past few years, we've been involved in multiple projects and have obtained a firsthand view of the issues and challenges that mobile application developers must face with networking analysis, debugging, and profiling mechanisms. Today, there are smarter devices for which we need smarter tools and solutions. Our ongoing work aims to simplify development of mobile applications by providing communication awareness so that applications can exploit all the intelligence available in these new smarter phones.

We're working to provide an application-level framework for isolating networking issues, which developers can incorporate in their applications. We're also integrating SymPA and QoSMeT,[8] a passive measurement tool for measuring one-way end-to-end network QoS developed by the VTT Technical Research Centre. Integrating these tools will enable one-way mobile device measurements in real time.

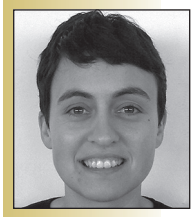
ACKNOWLEDGMENTS

The Spanish government-sponsored project TIN 2008-05932 and the Autonomous Community of Andalusia project P07-TIC-03131 funded this research.

REFERENCES

1. F.H.P. Fitzek and F. Reichert, *Mobile Phone Programming and Its Application to Wireless Networking*, Springer, 2007.
2. A. Ocampo et al., "Toward a Reference Process for Developing Wireless Internet Services," *IEEE Trans. Software Eng.*, vol. 29, no. 12, pp. 1122–1134.
3. A. Diaz et al., "Experimental Analysis of Peer-to-Peer Streaming in Cellular Networks," *Proc. Int'l Conf. Advanced Information Networking and Applications* (AINA 07), IEEE CS Press, 2007, pp. 784–791.
4. A. Diaz et al., "Evaluating Video Streaming over GPRS/UMTS Networks: A Practical Case," *Proc. IEEE 65th Vehicular Technology Conf.* (VTC 07), IEEE Press, 2007, pp. 624–628.
5. D. Wood, *Symbian for Software Leaders: Principles of Successful Smartphone Development Projects*, Symbian Press, 2006.
6. T. Zhang et al., "Reducing Energy Consumption on Mobile Devices with WiFi Interfaces," *Proc. Global Telecommunications Conf.* (Globecom 05), IEEE Press, 2005, vol. 1, pp. 561–565.
7. *User Equipment (UE) Radio Transmission and Reception (FDD)*, tech. specification TS 25.101, 3rd Generation Partnership Project (3GPP), 2008.
8. J. Prokkola et al., "Measuring WCDMA and HSDPA Delay Characteristics with QoSMeT," *Proc. IEEE Int'l Conf. Communications* (ICC 07), IEEE Press, 2007, pp. 492–498.

the AUTHORS



Almudena Díaz is a researcher at the University of Málaga. Her research interests are communication protocols, wireless networks, and mobile application development. Díaz has an MS in telecommunication engineering from the University of Málaga. She's a member of the IEEE and ACM. Contact her at almudiaz@lcc.uma.es.

Pedro Merino is an associate professor at the University of Málaga. His research interests are foundations, tools, and applications of formal methods for critical systems, particularly communications software and development techniques for Internet services on mobile networks. Merino has a PhD in computer science from the University of Málaga. He's a member of the IEEE Communications Society and ERCIM. Contact him at pedro@lcc.uma.es.

F. Javier Rivas is a PhD candidate at the University of Málaga. His research interests are mobile computing, software verification, and wireless communications. Rivas has an MS in telecommunication engineering from the University of Málaga. Contact him at franciscojavier.rivas@gmail.com.