Unleashing Shared-Experience Communications in a Mobile World

Richard Hull
Bell Labs Research
Alcatel-Lucent
Murray Hill, NJ 07974, USA
hull@research.bell-labs.com

Recent years have brought several dramatic advances in mobile services, including data and web services, location- and presence-based services, mash-ups, and rich improvements in mobile handsets and their interfaces. But the user's control and experience of shared and real-time communication services remains essentially unchanged. For example, audio calls, streaming video, and Instant Messaging are generally supported by silo technologies in the underlying network(s). With currently available technology, the creation of services that blend these capabilities and incorporate them into broader applications typically requires familiarity with a plethora of standards and protocols at various levels of abstraction. In the Session Data Types (SDT) project at Bell Labs, we are developing a novel approach for allowing web-level and other application programmers to incorporate shared-experience communication services into mobile (and other) applications, while insulating the programmers from the heterogeneity and intricacy of the telecommunications standards and protocols.

The central premise of the Session Data Type (SDT) framework is to treat a shared-experience communication session as a rich state machine. An API, which is currently supported through Java RMI but will be supported through WSDL and HTTP (and possibly SOAP), provides access to the current state of the session along with events from the network, including in-band signaling. Applications can query the state, can directly invoke transitions in the state machine, can respond to events explicitly, or can dynamically load event-handling rules into the session management infrastructure in the network. The transitions and event handling can include adding/dropping participants from (sub-)sessions, and launching, merging, and splitting of (sub-)sessions. Applications can also respond to events by invoking other web and communication services.

We believe that the SDT framework can be useful in combination with other, complimentary mechanisms that expose mobile network services, including end-user presence and location, personalization capabilities, and end-user profile data. Blended services using SDTs can be created using programming languages such as Java, high-level web-services languages such as BPEL, and web scripting languages such as AJAX. More broadly, we expect that that the conceptual simplicity of the SDT framework will enable a substantial increase in the creativity around blending of traditional web services and transactional mobile services (e.g., location, presence) with shared-experience communication services.

Admittedly, abstraction frameworks such as Parlay/ParlayX [1] already provide some of the benefits we anticipate from the SDT framework. One key difference between SDTs and Parlay/ParlayX is that the SDT framework is based on an explicit state-transition system for the communication sessions, applications can query that state at any time, and applications can subscribe for notifications using a relatively rich predicate language. This permits a clean separation of responsibilities between SDT and application, which appears especially useful in contexts where multiple media are involved, and in contexts involving a rich session structure with multiple subsessions. In particular, the complete information and management of such "rich" sessions can be handled by SDTs, whereas when using Parlay/ParlayX the application will have to explicitly manage different subsessions of the rich session as individual Parlay/ParlayX sessions. In terms of specific features not supported by the current ParlayX standard, the SDT framework enables applications to respond to in-band signalling, to merge sessions, to shift callers between sessions and subsessions, and to load triggered rules into the network.

The work on the SDT framework is on-going; a brief overivew of the project as of October, 2007, is available in [2]. Because the SDT API is based on state machines, we expect that it will provide an excellent basis for static analysis of applications that use SDTs. Similarly, the explicit use of session state should make it relatively easy to use SDT-based services in the context of semantic web services [3], thereby enabling automated discovery and composition of shared-experience communication services.

Acknowledgements This work is joint with Al Aho of Columbia University, and Bob Arlien, Glenn Bruns, Dennis Dams, Rob Dinoff, John Letourneau, Kedar Namjoshi, Frans Panken, and H. van Tellingen of Bell Labs.

References

[1] The Parlay Group, "Parlay group home page," `http://www.parlay.org/en/index.asp`.
[2] A. Aho, G. Bruns, D. Dams, R. Hull, J. Letourneau, K. Namjoshi, F. Panken, and H. van Tellingen, "Session Data Types: An abstraction layer for shared-experience communications in converged applications," in *Proc. 11th Intl. Conf. on Intelligence in Service Delivery Networks (ICIN)*, October 2007.
[3] S. McIlraith, T. Son, and H. Zeng, "Semantic web services," *IEEE Intelligent Systems*, vol. 16, no. 2, pp. 46–53, Mar-Apr 2001.