2014 Eighth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing

Towards Implementing Web-based Adaptive Application Mobility using Web Real-Time Communications

Dan Johansson
Department of Computer Science,
Electrical and Space Engineering
Luleå University of Technology
Skellefteå, Sweden
Email: dan.johansson@ltu.se

Mikael Holmgren
Department of Computer Science,
Electrical and Space Engineering
Luleå University of Technology
Skellefteå, Sweden
Email: mikael.holmgren@ltu.se

***Abstract*—Application mobility is a subcategory of service mobility, being defined as the process of migrating applications between different hosts during application execution. In this paper we present a decentralized architecture for web-based adaptive application mobility, using the WebRTC framework. To the best of our knowledge, our proposal is the first of its kind within the area. It is our purpose to show the feasibility of a decentralized system for web-based adaptive application mobility, and present WebRTC as a strong facilitator of that idea.**

I. Introduction

Application mobility is defined as the process of migrating applications between different host devices during application execution [1]. Most projects in this field (e.g. [2], [3], [4], [5], [6], [7], [8], [9], and [10]) target the migration of native applications between stationary devices. However, with the emergence of smartphones, the usage of applications targeted for mobile platforms (often referred to as apps) has become widespread, not least in settings where mobility is emphasized.

There are two main categories of apps: native apps and web apps [11]. Native apps are developed for specific platforms (i.e. a specified set of versions of a particular OS), using a programming language supporting that particular platform. Web apps are developed using web technologies, preferably HTML5 and related frameworks. While native apps in general can take better advantage of device hardware and OS features, web apps allow cross-platform functionality. In theory, a web app can be run on any device equipped with a web browser and a network connection.

Web Real-Time Communications, or WebRTC [12], is an open source framework supported by Google, Mozilla, and Opera, with the goal of delivering RTC capabilities via JavaScript to web browsers. The main architecture consists of a WebRTC C++ API directed towards browser developers, and a Web API aimed for web app developers. The web API [13] provides the establishment of peer-to-peer connections between nodes (i.e. browser-browser communication) using NAT-traversal technologies to send and receive streams and data directly between peers. A media stream interface called getUserMedia allows access to device hardware such as camera and microphone, without the need for proprietary plug-ins. WebRTC has been implemented as a facilitator of many different systems, from content distribution networks [14], accessibility enablers [15], conferencing systems [16], and multimedia distribution [17].

In this paper we present a decentralized architecture for web-based adaptive application mobility, and implementation details using the WebRTC framework. To the best of our knowledge, our proposal is the first of its kind within the area. It is our purpose to show the feasibility of a decentralized system for web-based adaptive application mobility, and present WebRTC as a strong facilitator of that idea.

II. Related projects

MDAgent [9] is a system for application mobility based on mobile agents that follow the user. A context manager keeps record of agent life cycles and store context data, that the applications can use to adapt to the environment. The system can be deployed on mobile devices. Implementation details are not available, but the deployment setup implies an intranet setting, thus making the system small scale.

The OPEN Migration Service Platform (MSP) [10] uses a central server to orchestrate application migration. The central node also manages context, state adaption and event handling. An important outcome of their VLAN testbed experiments is that migration latency increase almost linearly when using simulated link delays of 100 ms or higher. This pinpoints one of the qualitatively different challenges when designing for application mobility in settings with mobile devices and decentralized system setup, where no central node can be responsible for adaptation and byte code transfer.

XAM [18] combines both a decentralized layout and mobile devices, carrying out application migration within a peer-to-peer network allowing nodes from different networks communication over heterogeneous network technologies to join and exchange applications. The network is based on native technology, and experiments show that migration latency rockets when using cellular technology to migrate multi-MB-sized applications.

These are just a handfull of examples of related projects, but none of the related projects we have surveyed target application migration between different networks in mobile

978-1-4799-4331-9/14 $31.00 © 2014 IEEE
DOI 10.1109/IMIS.2014.60

settings using web technology. Our web-based approach follows a natural evolution of web technologies closing in on its native counterparts in terms of functionality, and the fact that smartphone application usage grows with an ever increasing speed.

III. Implementing Web-based Adaptive Application Mobility

In this section, we provide our architectural considerations and a system description for implementing web-based adaptive application mobility. We also present some initial results of our early prototyping.

A. Architectural considerations

An architecture for web-based adaptive application mobility was presented in [19]. The paper contained a theoretical framework connecting HTML5 functionality to the requirements of application mobility. The conclusions in that paper were that the emerging HTML5 standard along with related frameworks provided an environment to deliver adaptive web-based application mobility, and that the proposed architecture strongly met the requirements defined for that particular type of mobility. One weakness with the proposed architecture was however that it included a centralized approach, using a migration server. A centralized system layout has a higher degree of network dependency, is vulnerable to the loss or removal of key nodes, and, additionally, adding/removing nodes is more complicated compared to a decentralized system [18]. A decentralized system also capitalizes on the fact that end-objects are smart and capable [20]. We therefore proceed from the original architecture proposed in [19], but redesign it so it becomes decentralized. The key tool for making this possible is the emergence of WebRTC.

Besides a decentralized layout, our architectural goals are to design for robust application distribution and identification, context-awareness support, and provision of seamless migration. The system must also have cross-platform functionality, i.e. be able to migrate apps between different networks and onto different host devices. These goals are derived from research concerning application mobility requirements [21].

B. System implementation

WebRTC is the backbone in our decentralized system implementation. It provides the network with basic functionality regarding the addition and removal of nodes (i.e. available host devices) within the system. One or several devices in the network take on the role as server nodes, running scripts necessary for the formation of the peer-to-peer network. In our implementation we use node.js [22] along with the HTML5 Web Sockets API (socket.io) to accomplish this. Node.js is an event-driven web server based on javascript. The web server allows clients to find each other and interconnect.

Problems associated with node location [23] are solved through the peer-to-peer setup. A peer connection object is created, using a stun server[1] to overcome NAT problems. This makes it possible for nodes to be located outside the subnet. A connection request is created using the `createOffer` function provided through the WebRTC framework. The information provided by the client is combined with the information from the STUN server, creating a local description of the connecting node. When a client connects to the node.js server, its IP address is saved. The client can then send out offer messages to other clients connected to the server. If the server receives an answer message (containing similar descriptive information about the answering node – this is called a remote description), it will contact the client originally signaling the offer, whereupon the two clients can connect. As new devices are identified and subsequently added to the network, they can start communicate directly between each other, sending context updates and initiating application migration processes. Using disconnect messages, nodes can easily be removed from the peer-to-peer network. The client IP adress is consequently removed from the node.js list of available nodes.

[1]In our implementation we use a public test server provided by Google

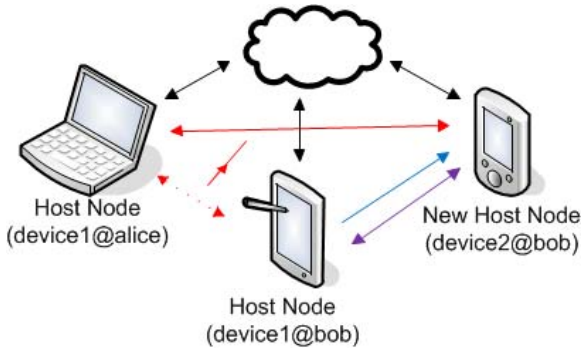

Fig. 1. System overview

Each device runs a Migration Manager (MM), responsible for initiating and receiving migration requests. The MMs are, along with all the migratable applications in our system, developed using web technology (i.e. HTML5, CSS3, and JavaScript), thus defining them as web apps. After connecting to the node.js server, this component is no longer needed. All additional migration functionality is added through the WebRTC framework.

The WebRTC peer connection function uses a RTC `DataChannel` interface to set up a bi-directional channel between the browser on the host device (device1@bob in figure 1) and the browser on new host device (device2@bob in figure 1). The clients are now ready to listen to `onMessage` events.

The migratable applications adapt to different host devices through the concept of responsive web design [24], using feature detection libraries to obtain information about both restraints and supported features of the heterogeneous platforms. The media stream interface included in WebRTC allows the applications to access device hardware, which in turn enables functionality such as multimedia retrieval or video conferencing. The Cache API, included in HTML5, gives the app developer an opportunity to decide which parts of an app (if not the whole app) should be downloaded and stored on the host device.

Application dependent data is stored locally, using HTML5 local db. During a migration process, the execution of the app is paused, and important app data stored as JSON objects, sent together with app code to the new host, where the states of

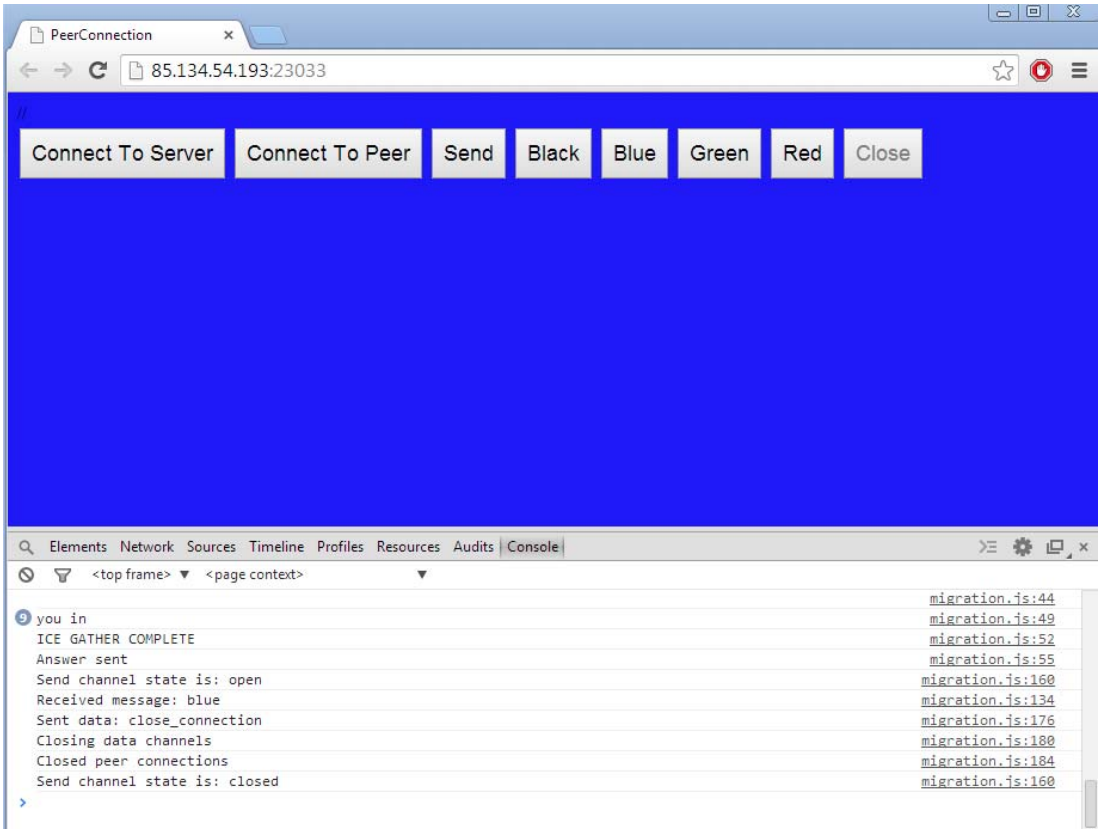

Fig. 2. Application GUI

the now migrated app are restored.

Figure 1 depicts signaling at the peer-to-peer network setup (black arrows). Migration proposals and negotiation is carried out directly between MMs (purple arrow). As an application is migrated (blue arrow), the eventual session is correspondingly re-established (red arrows).

C. Results

Although the architecture and the technical specifications are ready, as per now, the actual implementation is only partially realized. Finalization of the system implementation is ongoing, and therefore we can present initial results.

The system was set up on a Windows server 2008 R2 Enterprise, with a Pentium(R) Dual Core CPU E6600 @ 3.06GHz processor with an installed memory of 4 GB, running a 64 bit OS.

To test the system, a simple application is created. Using HTML5 and CSS, a web app is designed, equipped with a switch-background color feature. Although simple, this functionality gives us a state to go with the application, proving migration of more than byte code between clients. The application has a total size of 6 kB. Figure 2 shows screenshots of our rudimentary prototype GUI.

The actual migration is carried out in three steps: first, the two nodes connect to the node.js server. This is done manually in our prototype implementation through buttons in the user GUI. Their IP addresses are added to the server and simultaneously signaling to the system that they want to subscribe to all incoming connection offers. Another button in our prototype GUI allows a client to connect to another client; an offer is broadcasted to all clients in the peer-to-peer network, and as soon a client answers, a message will be sent back to the first client, and a peer-to-peer connection will be established. By clicking a third button (Send), the application will be migrated from the original host to the new host, maintaining all states. The cache is automatically emptied at the original host, making sure that there is only one copy of the application. Timestamps are set for the different events, to aid prototype testing.

In our first round of tests, only application states were migrated. The system proved fully capable of providing the application with current color data, at the same time closing down the web app at the original host.

In another set of test rounds, we used an additional windows server located abroad. The application states were successfully migrated between different networks, one located in Sweden and the other one in Finland.

Additional testing was carried out using a mobile device as host node, more specifically a Samsung Galaxy S3 4G running Android 4.3. Full state migration sequences were carried out, with no apparent errors. Although being of small scale, our additional tests showed that the prototype system supports the migration of application states also in mobile settings.

IV. Conclusions

We have presented a decentralized architecture for web-based adaptive application mobility, using the WebRTC framework as a core technology in our implementation. As for now, WebRTC is supported in the latest versions of web browsers such as Google Chrome, Mozilla Firefox, Opera, and it's mobile counterparts, and as more systems are developed using WebRTC, it is our belief that the framework eventually will become supported by most modern browsers, thus enabling exposure to WebRTC-based services, such as our implementation of application mobility, on a larger scale.

The main reason behind our work was not to compare the system to existing solutions based on other technology paradigms, but rather to prove the validity of the concept of web-based adaptive application mobility. Through our initial implementation, we have proved that HTML5 and related frameworks can be used to deliver migration of application states between both stationary and mobile nodes, within the intranet or between heterogeneous networks. Our main remaining target is to add the final functionality of byte code migration, realizing full application mobility based on state-of-the-art web technology.

Implementation is still in progress, and future work will consist of evaluation and more detailed measurements, as well as user studies on our prototype system in a real world setting, using heterogeneous networks and devices. We also plan to compare our web-based prototype with native counterparts, for instance our own java-based implementation of a system providing application mobility [18], using both quantitative and qualitative measurement variables.

Acknowledgment

The authors want to thank Samuel Engström, Dan Erikson, Daniel Henriksson, Mattias Jelbring, Alexander Lönnefelt, Phi-Long Vu, Pontus Werme, and Albert Öhrling for valuable programming efforts. Our thanks also go to Julia Bergstedt for additional help with testing.

This work was supported by the NIMO project, funded by EU Interreg IVA North program (see nimoproject.org).

References

[1] T. Koponen, A. Gurtov, and P. Nikander, "Application mobility with hip," in *Proc. of ICT'05*, ICT'05, 2005.

[2] R. Bandelloni and F. Paternò, "Flexible interface migration," in *Proceedings of the 9th international conference on Intelligent user interfaces*, IUI '04, (New York, NY, USA), pp. 148–155, ACM, 2004.

[3] Y. Zhou, J. Cao, V. Raychoudhury, J. Siebert, and J. Lu, "A middleware support for agent-based application mobility in pervasive environments," in *Proceedings of the 32nd International Conference on Distributed Computing Systems Workshops*, (Los Alamitos, CA, USA), p. 9, IEEE Computer Society, 2007.

[4] A. Ranganathan, C. Shankar, and R. Campbell, "Application polymorphism for autonomic ubiquitous computing," *Multiagent Grid Syst.*, vol. 1, pp. 109–129, Apr. 2005.

[5] H. Schmidt and F. J. Hauck, "Samproc: middleware for self-adaptive mobile processes in heterogeneous ubiquitous environments," in *Proceedings of the 4th on Middleware doctoral symposium*, MDS '07, (New York, NY, USA), pp. 11:1–11:6, ACM, 2007.

[6] P. P. L. Siu, N. Belaramani, C. L. Wang, and F. C. M. Lau, "Context-aware state management for ubiquitous applications," in *Embedded and Ubiquitous Computing*, EUC '04, pp. 776–785, 2004.

[7] T. Hwang, H. Park, and J. W. Chung, "Desktop migration system based on dynamic linking of application specific libraries," in *Advanced Communication Technology, 2006. ICACT 2006. The 8th International Conference*, vol. 3, pp. 1586–1588, feb. 2006.

[8] I. Satoh, "Self-deployment of distributed applications," in *Scientific Engineering of Distributed Java Applications* (N. Guelfi, G. Reggio, and A. Romanovsky, eds.), vol. 3409 of *Lecture Notes in Computer Science*, pp. 48–57, Springer Berlin / Heidelberg, 2005.

[9] P. Yu, J. Cao, W. Wen, and J. Lu, "Mobile agent enabled application mobility for pervasive computing," in *Proceedings of the Third international conference on Ubiquitous Intelligence and Computing*, UIC'06, (Berlin, Heidelberg), pp. 648–657, Springer-Verlag, 2006.

[10] K. Hojgaard-Hansen, H. C. Nguyen, and H. Schwefel, "Session mobility solution for client-based application migration scenarios," in *Proceedings of the Eighth International Conference on Wireless On-Demand Network Systems and Services*, WONS, pp. 76–83, jan 2011.

[11] A. Charland and B. Leroux, "Mobile application development: web vs. native," *Commun. ACM*, vol. 54, pp. 49–53, May 2011.

[12] Google, "Webrtc," Oct 2013. [Online]. Available: http://www.webrtc.org/.

[13] W3C, "Webrtc 1.0: Real-time communication between browsers," Aug 2013. [Online]. Available: http://dev.w3.org/2011/webrtc/editor/webrtc.html.

[14] L. Zhang, F. Zhou, A. Mislove, and R. Sundaram, "Maygh: building a cdn from client web browsers," in *Proceedings of the 8th ACM European Conference on Computer Systems*, EuroSys '13, (New York, NY, USA), pp. 281–294, ACM, 2013.

[15] J. Bose and K. Dipin, "A solution for a mobile computing device along with supporting infrastructure for the needs of illiterate users in rural areas," in *India Conference (INDICON), 2012 Annual IEEE*, pp. 519–524, 2012.

[16] A. Amirante, T. Castaldi, L. Miniero, and S. Romano, "On the seamless interaction between webrtc browsers and sip-based conferencing systems," *Communications Magazine, IEEE*, vol. 51, no. 4, pp. 42–47, 2013.

[17] L. Lopez Fernandez, M. Paris Diaz, R. Benitez Mejias, F. Lopez, and J. Santos, "Kurento: a media server technology for convergent www/mobile real-time multimedia communications supporting webrtc," in *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2013 IEEE 14th International Symposium and Workshops on a*, pp. 1–6, 2013.

[18] D. Johansson, K. Andersson, and C. Åhlund, "Supporting user mobility with peer-to-peer-based application mobility in heterogeneous networks," in *Proceedings of the 38th IEEE Conference on Local Computer Networks Workshops (LCN Workshops) : 7th IEEE Workshop On User MObility and VEhicular Networks*, 2013.

[19] D. Johansson and K. Andersson, "Web-based adaptive application mobility," in *Proceedings of the 1st IEEE International Conference on Cloud Networking*, pp. 87–94, 2012.

[20] J. L. Hernández-Ramos, A. J. Jara, L. Marín, and A. F. Skarmeta, "Distributed capability-based access control for the internet of things," *Journal of Internet Services and Information Security (JISIS)*, vol. 3, pp. 1–16, Nov 2013.

[21] D. Johansson, A. Åhlund, and C. Åhlund, "A mip-p2p based architecture for application mobility," in *Proceedings of the 10th International Conference on Mobile and Ubiquitous Multimedia*, MUM '11, (New York, NY, USA), pp. 85–93, ACM, 2011.

[22] Joyent, "Node.js," Feb 2014. [Online]. Available: http://nodejs.org/.

[23] V. P. Kafle and M. Inoue, "Locator id separation for mobility management in the new generation network," *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA)*, vol. 1, pp. 3–15, 9 2010.

[24] E. Marcotte, "Responsive web design," May 2010. [Online]. Available: http://alistapart.com/article/responsive-web-design.