Continuous Media Adaptation for Mobile Computing Using Coarse-Grained Asynchronous Notifications*

Dario Maggiorini, Daniele Riboni
University of Milano, Italy
{dario, riboni}@dico.unimi.it

Abstract

The recent spreading of public wireless infrastructures allowing for higher data rates makes mobile communications networks a very attractive platform for distribution of multimedia content. At the same time, limited resources in public wireless networks pose serious questions on how to bring services and multimedia to terminals to be used anywhere. Content adaptation is required in order to bring the best perceptual experience to the end-user while optimizing resources usage. Unfortunately, content adaptation is very difficult to achieve and is usually related to bandwidth availability only. In this paper we propose to extend existing service provisioning architectures with an asynchronous notification system to keep up-to-date the whole set of user profile data during service provisioning. We argue that the average multimedia application behavior, still adhering to a model based on a very limited number of choices, is not affected by increased reaction time and coarse-grained parameters responsivity. Furthermore, introduction of asynchronous notifications will enable service providers to adapt content considering any parameter characterizing the user profile, not just available bandwidth.

1. Introduction

With the wide availability of public wireless infrastructures allowing broadband access, the distribution of multimedia content is gaining momentum. Being bandwidth availability very limited in wireless networks, an optimal usage of resources is desirable in order to deliver ”the best” possible service to the end user. Content adaptation can be exploited in order to optimize bandwidth usage with regard to available resources together with environmental conditions (e.g., device capabilities, connection technology, and user action context).

* This work has been partially supported by Italian MIUR (FIRB ”WebMinds” project N. RBNE01WEJT_005).

The vast majority of Internet services use a web infrastructure. This infrastructure is not capable to cooperate with the network to monitor resources but can be enriched with profiles describing user context. Profile data should include any information useful for offering a “better” response to a user request; i.e., the information characterizing the user, the device, the network infrastructure, and the content involved in a service request. In particular, the dynamic nature of some profile attributes must be carefully taken into account when dealing with mobile computing environment.

Let us now consider the general case of a user requesting streaming media content (e.g., a movie trailer). Generally, the user requests the streaming media by selecting a hyperlink on his (micro)browser. Then, the HTTP request headers are analyzed by the server application logic in order to determine the most suitable bitrate. Obviously, data that can be obtained through HTTP request headers depend on the browser; typically, this information includes the user agent, acceptable movie encodings, screen resolution, number of colors and few others. In particular, no information regarding the network context comes with the HTTP headers; hence, usually the service provider asks the user to explicitly tell his network bandwidth. Considering these data, the server application logic chooses the bitrate to be provided and redirects the user’s browser towards the proper video content.

This mechanism has a number of shortcomings. One of them consists in the coarse grained specification of the network context. Secondly, this approach lacks a mechanism for adapting the streaming provision to the user’s preferences and context, which seems to be a key requirement for adaptation in mobile and ubiquitous computing (see [11, 12, 9]). The framework we adopt in this paper tries to fulfill the above mentioned requirements using a distributed profile management, and policies to model the dynamics of some data. As discussed in [3], in our opinion this architecture improves other frameworks (e.g., [8, 5]) in which the adaptation is based solely on the integration of distributed context data.

Another very relevant issue in mobile computing adap-

tation is that part of the profile information can change very quickly. If we consider the case of user profile data we can note that, while some attributes do not change during a session (e.g., the device screen size), other information may change depending on device status (e.g., available memory, battery charge status), user interaction with the device or application (e.g., turning on or off a feature), and user behavior (e.g., change of activity). Data owned by the network operator are, possibly, even more unstable. Part of these data can be very valuable for offering a better adaptation to users of continuous media contents. Hence, a mechanism for monitoring profile data *during* continuous media provision is required; a possible solution to this problem is presented in Section 5.

2. A Support Architecture for Mobile Services Provisioning

Throughout this paper we will refer to a framework for supporting service adaptation in mobile environments presented in [2] and [3]. This framework is flexible enough to permit personalized provisioning of most nowadays services provided via a web infrastructure. In the adopted framework, service providers can declare policies in order to dynamically personalize and adapt their services considering profile data (e.g., *"if the device battery charge level is low, then provide a low bitrate"*). Moreover, users are allowed to declare policies that determine their preferences regarding streaming content considering the current context (e.g., *"if I am involved in an important meeting, then I want to receive video with no sound"*).

Profile data and policies are owned by various entities located in different logical and physical places. Three entities involved in the task of building an aggregated profile have been identified: the *user* and his devices, the *network operator*, and the *service provider*. Each entity is associated with a Profile Manager (called *UPM*, *OPM*, and *SPPM* respectively) devoted to manage profile information and policies. In particular: *i*) The `UPM` stores information related to the user and his devices, including, among other things, personal information, user preferences, context information, and device capabilities, as well as his policies; *ii*) The `OPM` is responsible for managing attributes describing the current network context (e.g., location, connection profile, and network status); *iii*) Finally, the `SPPM` is responsible for managing service provider proprietary data including information about users derived from previous service experiences and service provider policies. Figure 1 provides a general overview of the adopted architecture. The main steps involved in a typical service request are the following: In Step 1, a user issues a request to a service provider through his device and the connectivity offered by a network operator. In order to retrieve the profile information

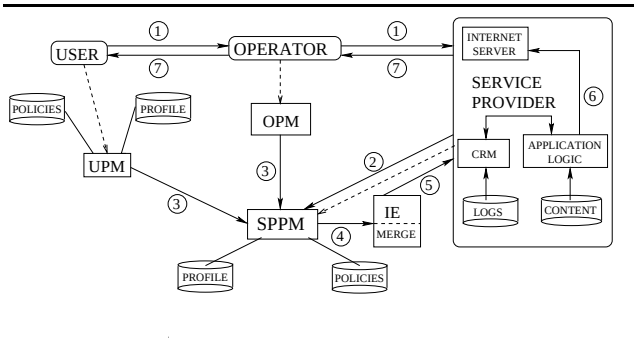

Figure 1. Information Flow upon User Request

needed to perform adaptation, the service provider queries the `SPPM` (Step 2), which in turn queries the `UPM` and the `OPM` to retrieve profile data and user's policies (Step 3). Then, in Step 4 the `SPPM` forwards collected and local profile data and policies to the Inference Engine (`IE`). In Step 5, the `Merge` module of the `IE` first aggregates profile data; then, the `IE` evaluates service provider and user policies against the merged profile, solving possible conflicts. The resulting profile attributes are then returned to the Service Provider, and are used by the application logic to properly perform adaptation (Step 6). Finally, in Step 7 the requested content is sent to the user.

3. Multimedia Adaptation

As outlined in Section 1, the dynamic nature of some profile attribute values claims for a mechanism for keeping up-to-date a relevant subset of profile data during the provision of the service. Unluckily, state of the art for dynamic adaptation of continuous media focuses on available bandwidth as the only parameter to be monitored for performing the content (re)encoding. In the last years many efforts have been devoted in devising techniques for bandwidth management at network and application level.

Network-level approaches can exploit explicit network feedback to monitor available resources, as described in [14], or request network cooperation to bound data transmission to a specific Service Level Agreement (*SLA*) as in the INT-SERV [7] and DIFF-SERV [6] architectures. The main disadvantage of these techniques is that nodes in order to operate in the network need to provide specific support for each given architecture and technology. This limits scalability and ease of deployment.

Application-level end-to-end monitoring for resource estimation has always been the most common approach to achieve service adaptation. Quantities of interest, especially available bandwidth, are estimated at the communication

endpoint observing packet dispersion [16, 18, 17, 13] either in probing traffic or existing transmissions. In any case, the application at client side is requested to cooperate, giving feedback to the server or, in some cases, to perform the estimation itself to improve the system scalability. The weakness of this kind of approach is in the estimation itself. Current *available bandwidth* techniques assume that the network is performing weighted fair queuing on its flows [15]. Available bandwidth measurements are known to follow multi-modal distributions [10] and therefore are especially difficult to measure and filter even in wired networks. In 802.11-based networks obtaining a reliable end-to-end measurement is questionable, especially for multi-hop configurations.

As a bottom line, with the currently deployed architectures all information for adaptation is held by the network operator and may lack of accuracy in some cases.

It is our opinion that many more parameters need be exploited in order to achieve content adaptation. We strongly believe that a lesser degree of precision in bandwidth estimation supported by availability of other attributes could contribute to a more flexible and easier strategy for multimedia adaptation.

4. Asynchronous Hi-Level Notifications

In order to include in the adaptation process the dynamics of network- and user-side data which can change during the continuous media provision, an asynchronous notification mechanism can be devised. A generic mechanism for asynchronous event notification can be obtained exploiting the triggers mechanism available in many DBMS. With regard to the network context, triggers can be considered as a lesser degree of explicit network feedback. Network feedback have been proved in [14] to be more reliable than end-to-end monitoring, especially on wireless networks.

Obviously, congestion plays a role in this case. Due to the limited rate of events the `IE` can handle, a much lesser degree of precision can be achieved by means of triggers compared to direct network monitoring. As a result we obtain delay in reaction and a lesser degree of freedom for content re-encoding. These two shortcomings proved to be no limitations when compared with application requirements.

Increased delay is not an issue since network overloading could result from transient traffic bursts that will be smoothed by buffers. On the other hand experiments with human subjects demonstrated that a momentarily loss of data with almost immediate recovery is preferable to a permanent loss of quality.

Loss of precision in monitoring, even when dealing with bandwidth, is not a disadvantage as well, since existing streaming systems rely on content re-encoding based on a limited number of combinations.

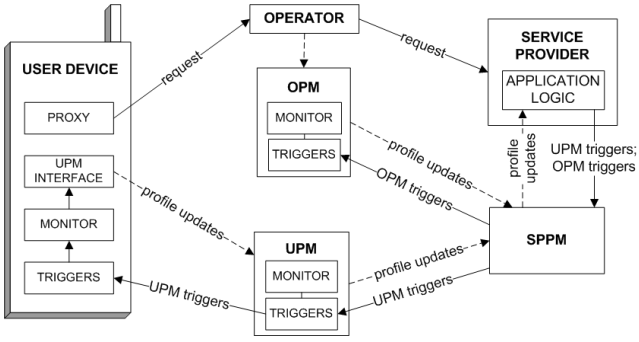

Figure 2. Trigger Mechanism

5. The Role of Triggers in the Architecure

In this section we describe in higher details a trigger mechanism for handling intra-session profile updates that has been presented in [1] and [4]. This mechanism allows to asynchronously notifying the service provider of changes in profile data on the basis of triggers. Triggers in this case are essentially conditions over changes in profile data (e.g., available bandwidth dropping below a certain threshold, or a generic change in the user's activity) which fire a notification when met. In particular, when a trigger fires, the corresponding profile manager sends the new values of the modified attributes to the `SPPM` module, which should then re-evaluate the policies.

Figure 2 shows an overview of the mechanism. To ensure that only useful update information is sent to the service provider, a deep knowledge of the service characteristics and requirements is needed. As a consequence, the conditions upon notifications are set by the service provider application logic, and communicated (step 1) to its `SPPM` module which appropriately forwards them to the `UPM` and to the `OPM` (step 2). Since most of the events monitored by trigger settings sent to the `UPM` are generated by the user device, the `UPM` communicates the settings to a server module (called *Triggers* in Figure 2) resident on the device (step 3). Note that, in order to keep up-to-date the information owned by the `UPM`, each client application must be equipped with an application monitoring the state of the device against the received triggers (called *Monitor* in Figure 2), and a client application (called *UPM Interface* in Figure 2) that updates the `UPM` when the *Monitor* notifies that a trigger fired. Each time the `UPM` receives an update that fires a trigger (step 4) it forwards the update to the interested `SPPM` (step 5). Finally, the integrated profile is re-computed, and changes in attribute values are communicated to the application logic (step 6), that can adapt the service.

In order to show the system behavior, consider the case of a streaming video service accessed through a Wi-Fi con-

nection. At first, the IE, evaluating the service provider policies determines a high bitrate since the OPM communicated that the access point is not congested. Hence, the service provider starts the video provision with a high bitrate. At the same time, the application logic sets a trigger to the OPM, asking a notification in case the available bandwidth drops below a certain threshold. Suppose that, during the video provision, the access point becomes congested. Then, the OPM sends a notification (together with the new value for the available bandwidth) to the SPPM, and the IE re-evaluates the rules. Since this time policy evaluation determined a lower bitrate, the video bitrate is immediately lowered by the application logic.

6. Conclusions and Future Work

In this paper we discussed the importance of continuous media adaptation in order to fulfill a user request in the best possible manner given the surrounding environment.

The shortcoming of existing techniques has been described: only available bandwidth is considered an interesting value and can be used as a parameter for re-encoding content. We argued about opening content adaptation parameters to user-provided information and policies as well as to a more scalable and easily deployable network interaction mechanism by means of asynchronous notifications. We strongly believe the coarse-grain imposed by asynchronous notifications is not a problem for streaming media application behavior, which is still adhering to an "*olympic*" model.

An existing service provisioning architecture will greatly benefit from an extension based on a triggering mechanism and distributed profile and policy management. This extension will increase the architecture scalability and bring the content adaptation model from mono-dimensional, based on bandwidth availability, to multi-dimensional and based on any parameter which can be useful for improving adaptation.

The implementation details of the proposed extension on an existing architecture and the integration with a video streaming system are under investigation. We plan to implement in a near future the prototype of an adaptive video server using asynchronous notifications.

References

[1] A. Agostini, C. Bettini, N. Cesa-Bianchi, D. Maggiorini, and D. Riboni. Integrated Profile and Policy Management for Mobile-oriented Internet Services. Technical Report TR-WEBMINDS-04, FIRB WEBMINDS, December 2003. http://webmind.dico.unimi.it/papers/TR03.pdf.

[2] A. Agostini, C. Bettini, N. Cesa-Bianchi, D. Maggiorini, and D. Riboni. Integrated profile management for mobile computing. In *Proceedings of Artificial Intelligence to Information Access Workshop (AI2IA), held in cooperation with IJCAI'03*, 2003.

[3] A. Agostini, C. Bettini, N. Cesa-Bianchi, D. Maggiorini, D. Riboni, M. Ruberl, C. Sala, and D. Vitali. Towards Highly Adaptive Services for Mobile Computing. In *Proceedings of IFIP TC8 Working Conference on Mobile Information Systems (MOBIS)*, 2004.

[4] C. Bettini and D. Riboni. Profile Aggregation and Policy Evaluation for Adaptive Internet Services. In *Proceedings of The First Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (Mobiquitous)*, pages 290–298. IEEE, 2004.

[5] M. Bowman, R. D. Chandler, and D. V. Keskar. Delivering Customized Content to Mobile Device Using CC/PP and the Intel CC/PP SDK. Technical report, Intel Corporation, 2002.

[6] R. Braden, D. Clark, M. Carlson, E. Davies, Z. Wang, and W. Weiss. An Architecture for Differentiated Services. *RFC 2475*, Dec. 1998. Work in progress.

[7] R. Braden, D. Clark, and S. Shenker. Integrated Services in the Internet Architecture: an Overview. *RFC 1633*, Jun. 1994. Work in progress.

[8] M. Butler, F. Giannetti, R. Gimson, and T. Wiley. Device Independence and the Web. *IEEE Internet Computing*, 6(5):81–86, September-October 2002.

[9] H. Chen, T. Finin, and A. Joshi. Semantic Web in the Context Broker Architecture. In *Proceedings of Proceedings of PerCom 2004*, 2004.

[10] C. Dovrolis, P. Ramanathan, and D. Moore. What do Packet Dispersion Techniques Measure? In *Proceedings of IEEE INFOCOM 2001*, Apr. 2001.

[11] C. Efstratiou, K. Cheverst, N. Davies, and A. Friday. An Architecture for the Effective Support of Adaptive Context-Aware Applications. In *Proceedings of the Second International Conference on Mobile Data Management (MDM 2001)*, pages 15–26, 2001.

[12] R. Hull, B. Kumar, D. Lieuwen, P. Patel-Schneider, A. Sahuguet, S. Varadarajan, and A. Vyas. Enabling Context-Aware and privacy-Conscius User Data Sharing. In *Proceedings of the 2004 IEEE International Conference on Mobile Data Management*, pages 187–198. IEEE, 2004.

[13] M. Kazantzidis, D. Maggiorini, and M. Gerla. Network Independent Available Bandwidth Sampling and Measurement. In *Proceedings* 2^{nd} *International Workshop on QoS in multiservice IP Networks*, Feb. 2003.

[14] M. Kazantzidis, I. Slain, T. Chen, Y. Romanenko, and M. Gerla. End-to-end versus Explicit Feedback Measurement in 802.11 Networks. In *The Seventh IEEE Symposium on Computers and Communications, ISCC02*, 2002.

[15] S. Keshav. Packet-Pair Flow Control, 1994. http://www.cs.cornell.edu/skeshav/papers.html.

[16] K. Lai and M. Baker. Measuring Bandwidth. In *Proceedings of IEEE INFOCOM '99*, Mar. 1999.

[17] K. Lai and M. Baker. Nettimer: a Tool for Measuring Bottleneck Link Bandwidth. In *Proceedings of the USENIX Symposium on Internet Technologies and Systems*, Mar. 2001.

[18] K. Lai and M. Baker. Measuring Link Bandwidths Using a Deterministic Model of Packet Delay. In *Proceedings of ACM SIGCOMM 2000*, Sep. 2000.