Chapter 5

Technological Solutions to Support Mobile Web 2.0 Services

Claudia Canali, Michele Colajanni, and Riccardo Lancellotti

Contents

The widespread diffusion and technological improvements of wireless networks and portable devices are facilitating mobile access to the Web and Web 2.0 services. The emerging Mobile Web 2.0 scenario still requires appropriate solutions to guarantee user interactions that are comparable with present levels of services. In this chapter, we classify the most important services for Mobile Web 2.0 and we identify the key functions that are required to support each category of Mobile Web 2.0 services. We discuss some possible technological solutions to implement these functions at the client and at the server level, and we identify some research issues that are still open.

5.1 Introduction

The so-called Mobile Web 2.0 originates from the conjunction of the Web 2.0 services and the proliferation of Web-enabled mobile devices.

The term Web 2.0, first introduced in 2004–2005 [20], indicates an evolution of the World Wide Web that aims to facilitate interactive information sharing, interoperability, user-centered design, and collaboration among users. Although it is difficult to precisely confine Web 2.0, its novel essence refers to a user-centric environment that is characterized by two predominant features [13]:

- Users may actively create and upload contents in many forms and have prominent profile pages, including heterogeneous information.
- Users may belong to a sort of virtual community determined by social interactions. For example, users may form connections among each other via explicit links to users who are denoted as "friends" or through "membership" groups of heterogeneous nature.

In the last years, we have also observed significant technological improvements in wireless networks and the diffusion of more powerful mobile devices with increased hardware and software capabilities. The growth and penetration of mobile communication technologies, with an expected number of global mobile phone subscribers reaching up to 4.5 billion in 2012 [19], has determined a scenario where users can access the Web directly from their mobile devices, and this trend is increasing. Mobile devices, which for portability reasons are companions in the user's daily life, are likely to become the favorite platform to connect, interact, and share content with other people.

The conjunction of Web 2.0 and mobile Web accesses is leading to a new communication paradigm, where mobile devices act not only as mere consumers of information, but also as complex carriers for getting and providing information, and as platforms for novel services [28].

We may expect that in the near future the demand for Web 2.0 services will mainly come from mobile devices [4,22]. This expectation is confirmed by the current trend of popular Web 2.0 sites, such as MySpace and Facebook, which offer mobile access to the users through specific applications preloaded on mobile devices. According to specialized studies, the most popular Web 2.0 sites are expected to have a mobile component within a few years [4].

Mobile Web 2.0 represents both an opportunity for creating novel services (typically related to user location) and an extension of Web 2.0 applications to mobile devices. The management of user-generated content, of content personalization, of community and information sharing is much more challenging in a context characterized by devices with limited capabilities in terms of display, computational power, storage, and connectivity. Furthermore, novel services require support for real-time determination and communication of the user position. The choice of appropriate technological solutions that can effectively support Mobile Web 2.0 services will be a key element to determine its success [17].

In this chapter, we propose a classification of Mobile Web 2.0 services and we evidence some key functions that are required for their support. We identify the main requirements for the implementation of each function. Finally, we discuss possible technological solutions for functions implementation at the client and server level, and identify some open issues.

This chapter is organized as follows. Section 5.2 proposes a classification of the emerging services for Mobile Web 2.0. Section 5.3 identifies the key functions required to support the Mobile Web 2.0 services. Section 5.4 describes some technological solutions to implement the supporting functions and identify possible open issues. Section 5.5 concludes with some final remarks.

5.2 Mobile Web 2.0 Services

Mobile Web 2.0 includes a wide range of heterogeneous and complex services. In this section, we propose a two-level classification of these services based on what we consider their predominant feature and other more specific characteristics. In the taxonomy shown in Figure 5.1, at the first level we have

- *Sharing services* that are characterized by the publication of contents to be shared with other users
- *Social services* that refer to the management of social relationships among the users
- *Location services* that tailor information and contents on the basis of the user location

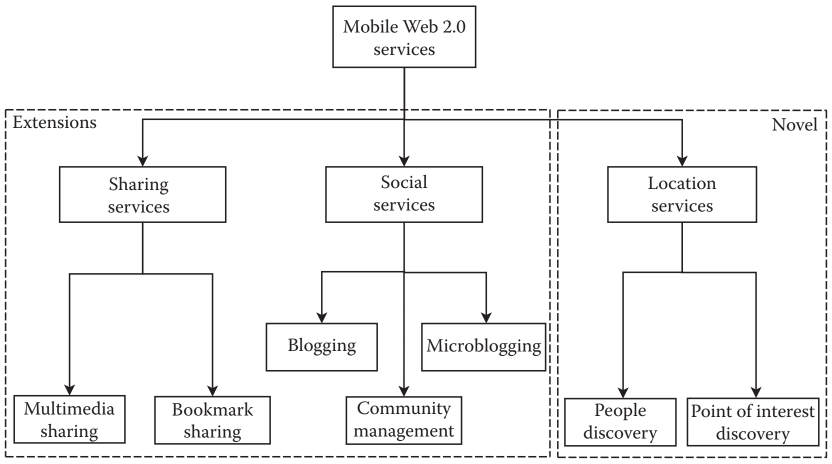

Figure 5.1

AQ1

As indicated by the dotted boxes in the figure, the sharing and social service classes represent extensions of existing Web 2.0 to the mobile scenario, while the location services represent a completely novel class of services, that exploits information on the user mobility.

It is worth noting that the service classes shown in Figure 5.1 as well as their subclasses, described below, are not completely disjointed categories.

5.2.1 Sharing Services

Sharing services offer the users the capability to store, organize, search, and manage heterogeneous contents. These contents may be rated, commented, tagged, and shared with specified users or groups that can usually visualize the stored resources chronologically, by category, rating or tags, or via a search engine.

The subclass of multimedia sharing considers management services related to multimedia resources, such as photos or videos. These resources are typically generated by the users that exploit the sharing service to upload and publish their own contents. Popular examples of Web portals offering a multimedia sharing service include Flickr, Zooomr, YouTube, Mocospace, and Mobimii.

The class of bookmark sharing services allows users to manage a common collection of Web page bookmarks. In this case, the shared contents are publicly available links to Web pages that users consider as interesting resources and want to share with other users. Many bookmarking services provide Web feeds for their lists of bookmarks, so that subscribers may become aware of new bookmarks as they are shared and tagged by other users. Among the most popular portals that offer bookmark sharing services we cite Del.icio.us, Reddit, Digg and its recently developed mobile version called Dgm8.

5.2.2 Social Services

The management of user relationships is the main feature of the social services that allow users to create social connections based on common interests, hobbies or experiences, and to actively interact with each other.

The services belonging to the *Community management* subclass allow registered users to maintain a list of contact details of people they know. Their key feature is the possibility to create and update a personal profile including information such as user preferences and his lists of contacts. These contacts may be used in different ways depending on the purpose of the service, which may range from the creation of a personal network of business and professional contacts (e.g., Linkedln), to the management of social events (e.g., Meetup), and up to the connection with old and new friends (e.g., Facebook, MySpace, Friendster).

The *Blogging* services allow a user to create and manage a blog, that is, a sort of personal online journal, possibly focused on a specific topic of interest. Blogs are usually created and managed by an individual or a limited group of people, namely *author(s)*, through regular entries of heterogeneous content, including text, images, and links to other resources related to the main topic, such as other blogs, Web pages, or multimedia contents. A blog is not a simple online journal, because the large majority of them allow external comments on the entries. The final effect is the creation of a discussion forum that engages readers and builds a social community around a person or a topic. Other related services may also include blogrolls (i.e., links to other blogs that the author reads) to indicate social relationships to other bloggers. Among the most popular portals that allow users to manage their own blog we cite BlogSpot, LiveJournal, Wordpress, and Splinder.

In *Microblogging* services, the communication is characterized by very short message exchanges among the users. Although this class of services originates from the blogging category, there are important differences between microblogging and traditional blogs: (1) the size of the exchanged messages is significantly smaller, (2) the purpose of microblogging is to capture and communicate instantaneous thoughts or feeling of the users, and (3) the recipient of the communication may differ from that of traditional blogs because microblogging allows authors to interact with a group of selected friends. Twitter, Jaiku, Plurk, Folkstr, GUSHUP, and Mobikade are examples of portals providing microblogging services.

5.2.3 Location Services

The ability to continuously trace user position represents one of the most innovative features in the context of Mobile Web 2.0, which emphasizes the important role of mobile devices in accessing the Web [26,31].

The knowledge of the user current location may be exploited in several ways to offer value added services. One of the most popular uses concerns *people discovery*, that basically aims to locate user friends; significant examples of this service may be found in Loopt, Brightkite, and Buddy Beacon applications. Usually these services, also called "friend finder" applications, plot the position of the user and his friends on a map; the geographical location of the users is uploaded to the system by means of a positioning system installed on the user mobile devices.

Another class of location services, that we call *points of interest (POIs) discovery*, exploits geographical information to locate POIs, such as events, restaurants, museums, and any kind of attractions that may be useful or interesting for a user. These services offer the users a list of nearby POIs selected on the basis of their personal preferences and specifications. POIs are collected by exploiting collaborative recommendations from other users that may add a new POI by uploading its geographical location, possibly determined through a GPS positioning system installed on the mobile device. Users may also upload short descriptions, comments, tags, and images or videos depicting the place. POI discovery services are provided by POIfriends, Socialight, and Mobnotes portals.

5.3 Functions to Support Mobile Web 2.0 Services

The previous section has pointed out that Mobile Web 2.0 includes complex and heterogeneous services, some totally new, others as extensions of existing Web 2.0 services. In this section, we identify some key functions that are at the basis of Mobile Web 2.0 services by separating functions that are required to extend Web 2.0 services to a mobile context from functions that are specifically related to the novel class of location services.

Among the functions required to extend Web 2.0 services to a mobile scenario, we identify

- Information input
- Large file upload
- Personalization
- Fruition of multimedia content

Other functions that are related to the possibility of localizing the mobile device are

- Computation of device location
- Geo-referenced information management

5.3.1 Functions Description

We give a brief description for each of the above functions to support Mobile Web 2.0 services.

The information input function refers to the communication of small size data, typically in a text format, from the users to the service through a mobile device.

Inserting comments in a blog or in a forum, tagging a resource, assigning ratings, updating personal information or simply adding a new entry in a microblogging service represent typical examples of information input. These operations occur very frequently in the Mobile Web 2.0 scenario, where users are not only consumers but also providers of information that actively interact with the services.

The *large file upload* function shares several common traits with the information input function; however, the focus is mainly related to multimedia resources, instead of textual information. The upload of user-generated large multimedia files (e.g., images, audio, and video resources) is a characteristic feature of Mobile Web 2.0 services that has been inherited by Web 2.0, and is becoming increasingly popular thanks to the diffusion of mobile devices equipped with built-in cameras.

The *personalization* function aims to tailor contents to the user preferences and needs [18]. Information about the users is collected by the services and may be exploited to offer personalized content in several ways, for example, through customized layout, information filtering, recommendation systems, subscription to specific channels or news feeds, and specification of lists of contacts.

Another feature of Mobile Web 2.0 that comes from Web 2.0 concerns the *fruition of multimedia contents*, which refers to the high demand for multimedia resources, such as images, video, and audio. Besides multimedia sharing services, also blogs, community management or POI discovery involve a considerable exchange of multimedia contents among mobile users.

The last two functions refer to the most innovative feature of Mobile Web 2.0 services, that is, the capability of identifying the current location of a mobile device. The geographical information may be exploited by the services to locate a user or a suggested point of interest. The *computation of device location* function is required to identify the geographical position of the mobile device. Once computed, the location data should be communicated to the service and stored as a geo-referenced data. The operations related to the storage and management of this data are accomplished by the *geo-referenced information management* function. It is worth to recall that a location data may refer both to a point of interest and to a user. If referred to a user, the current location may represent a volatile data that is subject to frequent updates.

5.3.2 Services and Functions

Each Mobile Web 2.0 service requires the support of at least one of the above functions. In Table 5.1, we map the functions and the service classes: each function is considered mandatory (*Yes*), not required (*No*) or optional (*Maybe*) for a specific service class. We do not include in the table the two functions related to the localization of mobile devices because their mapping on the services is quite straightforward and may easily be summarized as follows: computation of device location and geo-referenced information management are mandatory only for location services, but optional for all other services, where geo-referenced data may represent an additional information about users and uploaded contents.

Table 5.1 Functions to Support Mobile Web 2.0 Services

	Functions			
Mobile Web 2.0 Service	*Input*	*Upload*	*Personalization*	*Multimedia Fruition*
Multimedia sharing	Yes	Yes	Yes	Yes
Bookmark sharing	Yes	No	Maybe	No
Community management	Yes	No	Yes	Maybe
Blogging	Yes	No	Maybe	Maybe
Microblogging	Yes	Maybe	Yes	Maybe
People discovery	Yes	No	Yes	No
POI discovery	Yes	Maybe	Yes	Maybe

We observe that some input function is required by all the services through one or more operations, such as posting comments, adding ratings or tagging resources. The upload function is mandatory for multimedia sharing services, where the users upload self-generated large size files and where pictures taken on-the-fly may be added to enrich the short posts of the provided POI description; the upload function is optional in microblogging and POI discovery services. The personalization AQ2 function is optional for bookmark sharing and blogging services, where it may be exploited to filter contents and subscribe to news feeds, while is mandatory for all the other services that strongly rely on personal information maintained in the user profiles.

Although multimedia fruition does not represent a characterizing feature, it is mandatory in multimedia sharing services, while optional for community management, due to the nature of the exchanged contents, such as blogging, microblogging, and POI discovery services, where the presence of multimedia contents is possible. AQ3

5.3.3 Mobile Device Limitations

Supporting the previously described functions in a mobile scenario is not trivial due to the limited capabilities of mobile devices in terms of connection, CPU/storage capabilities, display size, and interface usability.

The bandwidth available to mobile devices has been greatly increased with the diffusion of 3-G wireless networks, and further enhancements are expected with the advent of 4-G technologies [11]. However, the wireless network connection remains relatively unstable and heterogeneous, because it is affected by coverage issues.

The computational power and storage capacities have experienced significant improvements in mobile devices; however, they remain significantly lower than those of a laptop/desktop that is typically used to access Web 2.0 services. The computational power and storage capacity may introduce problems when the mobile device must handle complex or multimedia-rich services, preventing the fruition and the storage of some kinds of resource formats [12]. Furthermore, computationally expensive functions consume a significant amount of energy, thus affecting the lifetime of batteries.

The display is characterized by minor improvements because of an intrinsic constraint of portability of the mobile devices. Even if the trend is toward devices with at least 3 inches screens and resolutions of 480 × 320 pixels [32], the limited display size makes for very unpleasant user navigation while accessing services designed for desktop computers. Finally, the device interfaces, which force users to insert input data through tiny keypads or small on-screen keyboards, are hardly satisfactory for the users who actively interact with the service.

Even if the technological evolution has substantially improved the scenario of the mobile devices population [28,32], the above limitations may hinder the deployment of the key functions required for Mobile Web 2.0 services. In Section 5.4, we describe some technological solutions that may be exploited to implement the functions to support Mobile Web 2.0 services, anticipating that the implementation of some functions presents open issues that deserve further research efforts.

5.4 Technological Solutions for Function Implementation

The main functions to support Mobile Web 2.0 services (information input, large file upload, personalization, fruition of multimedia contents, and location-related functions) may be implemented through different technological solutions. Each solution may follow a *client-side* or *server-side* approach depending on where the functions are implemented. The best approach is not an absolute choice, but it strongly depends on the specific context of each service, as discussed in the rest of this section. We can anticipate that solutions following a server-side approach may rely on more powerful platforms and usually do not present severe issues. However, in the Mobile Web 2.0 context many functions necessarily have to be implemented on the client side, even if this means coping with the limitations of the mobile devices.

5.4.1 Information Input

When the information input is carried out through a mobile device, the presence of intermittent or low bandwidth connections, as well as the small display and the peculiar input methods of these devices, introduce novel challenges that are not present in Web 2.0. For example, network connection quality can lead to poor navigation experience because tasks such as page refresh may be very slow with an unreliable and limited wireless connection. Furthermore, typing input information represents a cumbersome task in most mobile devices because the user must rely on a stylus or must interact with small on-screen keyboards. These limitations may result in an unsatisfactory experience that could move away users from these services. Allowing a comfortable navigation and supporting information input through a mobile device is a key issue in the context of Mobile Web 2.0, because this function is a fundamental element for all the offered services.

The problem of managing user input is usually addressed by client-side solutions and requires operating on two distinct elements: first, the communications between mobile devices and servers should be optimized to reduce the need for (synchronous) data exchange; second, the interface of the service should be redesigned to allow comfortable user navigation and input operations in a mobile device.

To cope with the first problem, the typical solution is to adopt asynchronous communications between the client and the server. This solution allows the browser on the mobile device to communicate with the server in the background in an asynchronous way with respect to the user interactions and without interfering with the current state of the page. The responses from the server are handled asynchronously by the browser that updates the Web pages without having to keep the user attention frozen. This benefit is particularly valuable in a mobile context, where the interaction with the service occurs while the user is on the run and cannot stay continuously focused on the device. By sending requests just for the required data that typically represent just a small portion of the information managed by the whole Web page, asynchronous communications between the client and the server allows to greatly reduce the amount of reloading and data transferred. Furthermore, the solution allows improving the user input speed and reduces the display processing requirements.

Typical examples of technologies supporting this approach are Asynchronous JavaScript and XML (AJAX) and Flex. AJAX is considered one of the most important enabling technologies for implementing interactive services. Actually, the term AJAX indicates a mixture of several technologies that integrate Web page presentation, interactive data exchange between client and server, client-side scripts, and asynchronous updates of server response [24]. Flex is an evolution of the widespread Flash technology commonly used to create animation, advertisements, and to integrate video into Web pages. Flex objects offer functionalities that resemble the AJAX approach, allowing the deployment of services that exploit asynchronous communication with the server to improve the user experience in the Mobile Web 2.0 scenario.

A limitation for asynchronous communications at the Web level in this scenario is that not all mobile Web browsers support the required technologies with adequate performance. While the most recent mobile devices satisfy this requirement, older devices with less computational power and memory may not be suitable to support asynchronous communication with the server. An increasingly popular alternative is to redesign the interface at the client level, without need to use Web browsers. In this case, the service is provided directly by an application installed in the mobile device, and interaction with the server is managed asynchronously, possibly using the same Web application programming interface (API) that are already AQ4 available for interaction through the Web browser. This approach, typically relying on the Java platform for mobile devices or through some device specific software development kit (SDK) (as in the case of the iPhone Objective-C API), has the AQ5 potential to reduce the computational and memory requirements that may hinder the popularity of a service accessed through a Web mobile browser. However, this approach requires a significant amount of effort for the development of the application. Furthermore, the client applications usually need frequent updates that in most cases have to be done manually.

The second critical problem is the difficulty for users to interact with user interfaces and type data for input in a mobile device. The need to rely on stylus or the small size of on-screen keyboards in mobile devices suggest that, even if asynchronous communication can improve the user experience, input operations remain a key issue for the diffusion of Mobile Web 2.0 services. The problem of supporting seamless input from user remains an open issue to address, although some possible research direction seems more mature and promising than others.

A first solution may be to redesign the user interface to avoid input whenever possible. The typical approach is to rely on fill-in forms that are pre-compiled based on default settings that may be personalized for each user. In this way, the user does not have to compile forms, but can simply choose between one or more options.

The alternative approach is to exploit client-side technologies to simplify the input operation by defining novel interfaces for human and computer interaction. To this aim, speech recognition and user gestures recognition (based on accelerometers) are gaining popularity in mobile devices [28]. The main drawbacks of this solution are the difficulty to integrate user interaction in a Web-based interface and the need to adapt these systems to mobile devices.

Alternative solutions may be too computationally expensive for current mobile devices. For example, speech recognition with a large vocabulary may be computationally unfeasible on CPU-power constrained devices. In a similar way, access to accelerometer for gesture recognition needs to be tailored to every specific device characteristic, thus hindering the adoption of general solutions. We expect that this area of alternative methods for human–computer interface will receive significant attention from researchers and industries, with the goal of simplifying the task of information input from the user thus enabling the development of even more sophisticated and interactive services.

5.4.2 Large File Upload

As for information input, the limitations of mobile devices may represent an issue that hinders the possibility for users to directly upload their self-generated contents, such as pictures and videos.

The user interface of a mobile device is often inadequate to support large file uploads to Mobile Web 2.0 services. For example, uploading a picture from a mobile device requires the off-line creation of the resource with the built-in camera, the temporary storage of the multimedia data in the device file system, and finally to seek the saved file from among the directories for the upload operation. While this behavior is acceptable when working on a PC or a laptop, it becomes unacceptable for the users of a Mobile Web 2.0 service. A user interface that simplifies the upload of a large file and especially multimedia content (for example, through a click-and-upload feature) will play a key role for the success of Mobile Web 2.0.

The issues related to the upload of large files are typically addressed at the client side. A solution is to exploit specialized clients that provide direct and easy upload of user-generated contents without the need to rely on Web-based upload forms or on multimedia messaging service (MMS). For example, specialized client applications AQ6 may allow pictures or videos to be taken directly through the device built-in camera and uploaded with a single click. This support has the potential to strongly affect the success of Mobile Web 2.0 services. A clear confirmation of this claim can be found in the recent announcement by Google: daily YouTube uploads directly from mobile devices have increased 400% in 6 days after the release of the last model of iPhone [30] that provides users with an easy interface to upload videos to the YouTube portal.

Another critical issue for uploading large files from a mobile device is related to the quality of the wireless connection. The upload time may increase to an unacceptable level, and, in the case of disconnections, upload may fail. The exchange of large amount of data may also reduce the battery lifetime and limit the possibility for the user to interact with the service. The problem seems to be even more critical if we consider the current trend of installing high resolution camera (up to several Mpixel per image) in mobile devices. To overcome this limitation, it is possible to carry out some content adaptation before the upload. For example, images can be cropped or scaled directly on the mobile device to reduce the amount of data transferred through the network.

However, the actual effectiveness of these solutions remains an open problem. Indeed, there is a trade-off involving computational power, network connection, and battery power. On one hand, content adaptation on the device requires a significant amount of computational power that may not be available on every device. On the other hand, transferring the high resolution resources without any adaptation to the server consumes network resources that in mobile devices may be scarce as well. Furthermore, both CPU-intensive operation and wireless data transfer have a significant impact on the mobile device batteries. The research for solutions that can address this trade-off, for example, combining client-side and server-side adaptation, represents an open issue that is likely to receive growing amount of attention in the next years.

5.4.3 Personalization

The personalization function requires an initial phase of user data collection, possibly from different sources. Then, the gathered information has to be stored in a user *profile* and maintained for subsequent use. The collection and the management of the user information are typically accomplished through a server-side approach.

The user information may be obtained basically from two sources:

- Explicitly communicated by the user
- Implicitly acquired from the user behavior

In the case of explicit communication by the user, personal information are provided through apposite fill-in forms to add/edit user preferences; this communication may occur when the user registers himself for the access to a service or may be filled/modified later.

When implicitly acquired, the user information is typically inferred through the analysis of the user behavior, for example, through data mining operations on a Web site log files or on sets of really simple syndication (RSS) feeds the user subscribed to [14]. Collaborative filtering techniques may also be exploited for group users based on similar preferences or click history [15]: missing information about a user may be integrated by considering the corresponding information in the profile of other users belonging to the same group. These techniques for implicit user profiling represent server-side solutions and are usually carried out off-line because they involve time-consuming operations, such as data mining. AQ7

The user profiles are usually maintained in database(s) on the server infrastructures. We should consider that the infrastructures to support Mobile Web 2.0 services typically consist of distributed systems with multiple servers, such as Content Delivery Networks [5]. Solutions for replicating data storages on a distributed infrastructure have been widely studied in the context of databases [21].

The simplest solution to manage database replication in distributed Web environments is based on a centralized master copy and replicated secondary copies. In case of updates, data are modified on the master copy and the changes are then propagated to the secondary copies. However, the access patterns for the user profiles present a unique feature that may help the management in case of replication. Specifically, each user typically interacts with only one server; hence the profile of a given user is accessed by one server for the whole duration of a user session. This access pattern has a significant impact on consistency and replication policies. Indeed, the whole dataset of user profiles can be partitioned and distributed over the servers depending on the user access patterns. Since no replication is needed, consistency issues are limited to guarantee that the user profiles on the servers are consistent with the data of the master copy.

User migration among multiple servers, however, may occur between consecutive sessions. Therefore, the user profile data should migrate following the user. The support for this behavior is not explicitly optimized in most replication strategies for back-end databases. An example of proposal to handle profile migration is Tuxedo [25], a generic data caching framework that supports user mobility by allowing data to follow the user.

It is worth to note that a unique opportunity offered by Mobile Web 2.0 concerns the use of *Subscriber Identity Module* (*SIM*), removable cards for personalization purposes. SIM cards have always been used to store data such as international number of the mobile user, billing information, security authentication and ciphering data, subscriber address books, etc. However, the advent of Mobile Web 2.0 has created the opportunity to exploit SIM cards as a place to store user authentication data for personalization purposes. This solution follows the philosophy of a "unified login" that allows users to log in to many Mobile Web 2.0 services using just one account, that is maintained by simply moving the SIM card from device to device.

The possibility to exploit a sort of unified login is particularly important to provide users with customized information coming from different Mobile Web 2.0 services. A typical behavior of Mobile Web 2.0 users, indeed, is to subscribe to several services and provide different information to each of them. For example, for each service a user may specify a list of contacts; to communicate or share contents with all his contacts, the user should separately access all the subscribed services. *Mash-up* technologies are increasingly being used to provide users with updated information coming from different subscribed services without the need of separately accessing all of them. The term mash-up indicates an approach that allows easy and fast service integration of data and functionalities from two or more external sources by using publicly available APIs. Thanks to this content aggregation, users may have available on a single page information coming from different services and updated in real time. Furthermore, mash-up solutions also allow integrating personal profile information maintained by different services without duplication of the information itself, thus simplifying update operations. Architecturally, the content aggregation usually takes place on the client side, by exploiting the Web browser of the mobile device to combine and reformat the data retrieved from multiple services.

5.4.4 Fruition of Multimedia Content

The poor connections of wireless networks and the reduced hardware capabilities of mobile devices may determine critical issues for the fruition of multimedia contents such as (1) the low and unreliable network bandwidth may cause long latency while downloading multimedia resources, (2) the computational power may be insufficient to decode and render high quality multimedia resources and (3) the small display size may not support high resolution formats. These limitations give rise to the need for adapting multimedia contents to match the capabilities of mobile devices and network connections.

Content adaptation may involve a wide range of heterogeneous transformations that are applied to the original contents to generate adapted versions suitable to be consumed by mobile device [9]. The basic idea behind content adaptation is that mobile users often do not need a best-quality experience when consuming multimedia resources, but rather a good-enough quality and acceptable latency to convey the needed information [34].

The adaptation is typically applied to multimedia resources with the main goal of reducing their size. Size reduction helps to decrease downloading time and storage requirements, and may also reduce, depending on the type of adaptation, the computational demand to render the resource on the mobile device. A large and heterogeneous set of resource attributes can be considered for each type of multimedia resource to perform the adaptation [9]. For example, image adaptation typically includes scaling, cropping, or compressing the image; audio resources are adapted by reducing the bit rate [16]; common transformations for video resources are frame size and color depth reduction [16].

The adaptation of multimedia contents may be performed at the client side or at the server side.

A critical aspect of content adaptation is the high computational cost of the transformations, especially when applied to large-sized multimedia resources. The computational cost of content adaptation may easily exacerbate the capabilities of the supporting infrastructures [7]; for this reason, determining the platform where content adaptation should be carried out represents a strategic choice.

In a client-side approach, contents are adapted directly on the mobile device. The advantage of this solution is that the adaptation may generate a resource version that perfectly matches the device limitations, thanks to the exact knowledge that the device has of its capabilities. However, this approach is not always feasible or convenient due to the device limitations. The reduced storage, computational power, and battery energy may prevent performing locally the expensive adaptation tasks. Furthermore, a client-side solution does not address the issue of poor connections: since the multimedia resources have to be entirely downloaded on the mobile device, long latency may be experienced while transmitting the content over the wireless connection. Hence, we may observe that, even if mobile devices are becoming more powerful platforms with medium-large connections, their limitations still prevent relying only on client-side adaptations. The technological evolution, however, allows the mobile devices to consume larger size and better quality resources with respect to the past and, if necessary, to carry out locally some final adjustments on multimedia contents.

The server-side approach represents a more feasible solution where the content adaptation is carried out on the server infrastructure. In this case, there are two main alternatives about *when* multimedia resources should be adapted: *on-the-fly* and *off-line* adaptation.

If on-the-fly adaptation is applied, the server infrastructure generates an adapted resource version for the specific mobile device at the moment of the request. However, the high computational costs of adaptation may hinder the effectiveness of this solution in the Mobile Web 2.0 scenario. Solutions based on on-the-fly adaptation have been proposed in the past [27], usually integrated with caching strategies at the intermediary level [3,10]. However, this approach was feasible in a context characterized by a limited amount of available multimedia resources and a small fraction of requests coming from mobile devices and, consequently, requiring adaptation. On the other hand, an on-the-fly approach may lead to excessive computational costs for the server platforms in the Mobile Web 2.0 scenario, even if coupled with caching strategies.

The off-line approach consists in pre-generating multiple adapted versions of multimedia contents that are maintained on the server infrastructure or cached at an intermediary level and then delivered to the user when requested. Relying completely on off-line adaptation means to pre-generate adapted versions of all multimedia resources for any class of device/connection, thus avoiding the expensive cost of on-the-fly adaptation. Furthermore, the use of layered encoding technologies for the off-line generation of adapted versions allows achieving significant advantages. Layered encoding allows generating only one adapted version of the content from which it is possible to obtain a suitable version for any mobile device. Basically, this approach generates a base layer and one or more enhanced layers to achieve the desired resolution of the multimedia content. Layers may be added or dropped depending on the requirements of the mobile device. A popular technology for layered encoding is Scalable Video Codecs (SVC) [23], where the enhanced layers may add temporal and/or spatial quality to the base layer. The use of SVC provides important benefits from the computational and storage points of view for systems adopting off-line adaptation solutions for supporting Mobile Web 2.0 services. The original multimedia content, indeed, has to be encoded only once, and the result is a scalable adapted version from which representations with lower quality can be obtained by discarding parts of the data. This solution avoids the need of storing multiple versions of the same multimedia content to satisfy any possible combination of requirements of mobile devices and wireless networks, thus simplifying even the server-side approach to the content adaptation.

To this aim, we should consider that the technological evolution of the mobile devices may have positive consequences for the off-line approach, because resources will not need to be tailored exactly for every type of client device as it happened until now. For example, while the first generation of devices ranged from monochrome to full-color capabilities, modern devices can display at least 16-bit color images; hence previous adaptations from color to B/W videos are now useless. Thanks to the technological improvements, different devices are now able to consume the same version of a multimedia resource, thus reducing the number of adapted versions that must be generated for every original resource. On the other hand, the presence of user-generated content in the Mobile Web 2.0 scenario is causing an explosion of multimedia contents in terms of quantity and heterogeneity [1,8]. We should also consider that the working set of accessed multimedia resources in Mobile Web 2.0 is highly volatile. Indeed, the resources are characterized by a short life span, because they typically concern real-world events or hot topics for which user interest rapidly subsides. For these reasons, a pure off-line solution may be not feasible or convenient due to the excessive waste of storage and computational power caused by pre-generating and maintaining adapted versions for every multimedia resource.

A hybrid solution that combine on-the-fly and off-line content adaptation may represent a better choice to support Mobile Web 2.0 services. A possible solution consists in applying off-line adaptation only to a limited set of the most popular resources, while adapting on-the-fly the remaining resources [6]. The rationale behind this approach originates from the popularity of multimedia resources in Mobile Web 2.0, that follows a Zipf-like distribution [8,33]. This means that pre-generation of adapted versions for a limited fraction of popular resources allows a system for Mobile Web 2.0 services to satisfy a high number of user requests. For the remaining requests, adaptation may be applied on-the-fly without overcoming the capabilities of server infrastructures. However, identifying the most popular resources represents an open challenge especially in the context of Mobile Web 2.0, whose workload is characterized by high volatility, short resource life span, and sudden popularity peaks.

5.4.5 Location-Related Functions

The possibility to geographically locate a mobile device and exploit this information to enrich the user experience is one of the most innovative features of Mobile Web 2.0. Two main functions are needed to support this feature: computation of device location and geo-referenced information management.

5.4.5.1 Computation of Device Location

Several solutions may be exploited for positioning purposes, that is, determining the geographical location of a mobile device. Positioning is usually performed by following a client-side approach, where the computation of the location is carried out on the device, then communicated to the service.

The most popular positioning technology is the Global Positioning System (GPS), whose wide adoption is due to the large diffusion of mobile devices equipped with GPS receivers that provide reliable three-dimensional location (latitude, longitude, and altitude). However, relying on the GPS technology for positioning may present two main drawbacks: the long time taken by the mobile device during the start-up phase to look for available satellites (between 45 and 90 s on average), and the consequent considerable cost of computational and battery power.

To overcome these limitations, the assisted GPS (A-GPS) has been introduced in the last few years. A-GPS is a carrier network dependent system that can improve the initial performance of a GPS satellite-based positioning system. Basically, the A-GPS uses an assistance server that communicates to the mobile device information on the available satellites to accelerate the signal acquisition.

In specific conditions, such as in indoor environments, an alternative technique to provide positioning is to exploit cellular or Wi-fi triangulation, based on the device distance from cell towers or Wi-fi access points. It is worth to note that GPS- and triangulation-based technologies may also be employed together to improve positioning accuracy [31].

Although mobile devices have become more powerful, they do not always have the computational power necessary to compute their current location through GPS or triangulation techniques. We should also consider that the computational cost of the operation depends on the accuracy required by the specific service, and on the frequency/speed of the users' movements, that may cause frequent recomputations of the exact location. In this case, the computation may be executed on the server side. The server infrastructure receives from the mobile devices GPS- and/or cell-based information and calculates the location and transmits the result to the devices.

5.4.5.2 Geo-Referenced Information Management

The device location represents a geo-referenced data that has to be stored and managed on the server side. The presence of geo-referenced data requires the use of technological solutions that allow operations on spatial data. For example, the system should be able to find all the points of interest that are close to a given user location or to identify the shortest path among two given locations. This requirement may be addressed by specialized Geographic Information Systems (GIS) or by database systems that support the storage and management of spatial data.

However, this requirement does not represent an open issue, due to the wide diffusion of GIS technologies and database systems with spatial support (e.g., MySQL, PostgreSQL, Oracle, Microsoft SQL Server).

Another important characteristic for management purposes is the potentially dynamic nature of the device location data that may change frequently due to user movements. Traditional approaches for data replication are not suitable to store and maintain potentially dynamic user location due to consistency issues. For this reason, a commonly adopted solution is to maintain the user location at the application server level just for the duration of the current user session. On the other hand, location data referring to POIs may be stored in databases due to their more stable nature.

A last consideration about the management of geo-referenced information is that the current location is usually considered by the users as a sensitive data [29]. Hence, Mobile Web 2.0 services should provide users with appropriate mechanisms to control the disclosure of their location. This issue is typically addressed on the server side. A first solution consists in allowing the user to edit a list of authorized contacts that may access his location. The user must have the possibility of modifying/updating the authorization list that is maintained in the user profile on the server infrastructure, at any moment. This is particularly important in the context of people discovery services, because the users' movements are continuously tracked to communicate their presence to nearby contacts even when they are not actively interacting with the application. A more sophisticated approach to preserve the privacy of the user consists in revealing the user position with different accuracies, depending on the specific location and/or on the recipient of the information [2].

5.5 Conclusions

The popularity of Web 2.0 services, coupled with the diffusion of increasingly powerful Web-enabled mobile devices, has led to the advent of Mobile Web 2.0. This emerging scenario includes very complex and heterogeneous services: some services are totally new, based on the notion of user location, while others are extensions of existing Web 2.0 services to a mobile context. The deployment of new and extended services may be hindered by the limited capabilities of mobile devices in terms of display, computational power, storage, and connectivity. Hence, the choice of appropriate technological solutions is a key element to effectively support Mobile Web 2.0 services.

We classify the emerging Mobile Web 2.0 services and we identify some key functions required for their support, discussing possible technological solutions for the implementation of each function. We show that existing technological solutions are sufficient to implement most functions; hence the problem is a correct integration and capacity design. On the other hand, the implementation of other functions represents a challenge that deserves further research efforts, as in the case of providing comfortable interfaces for user input and large file upload.

References

1. Berg Insight AB. Mobile Internet 2.0. Research Report, May 2007.
2. E. Bertino. Privacy-preserving techniques for location-based services. *SIGSPATIAL Special*, 1(2):2–3, 2009.
3. S. Buchholz and T. Buchholz. Replica placement in adaptive content distribution networks. In *Proceedings of the 2004 ACM Symposium on Applied Computing* (*SAC'04*), pp. 1705–1710, Nicosia, Cyprus, March 2004.
4. BusinessWeek. Social networking goes mobile. May 2006.
5. C. Canali, V. Cardellini, M. Colajanni, and R. Lancellotti. Content delivery and management. In *Content Delivery Networks*, R. Buyya, M. Pathan, and A. Vakali (eds.), pp. 105–126. Springer, Berlin, Germany, 2008.
6. C. Canali, M. Colajanni, and R. Lancellotti. Resource management strategies for mobile web-based services. In *Proceedings of 4th IEEE International Conference on Wireless and Mobile Computing* (*WIMOB'08*), pp. 172–177, Avignon, France, October 2008.
7. C. Canali, M. Colajanni, and R. Lancellotti. Performance evolution of mobile-web based services. *IEEE Internet Computing*, 13(2):60–68, March/April 2009.
8. M. Cha, H. Kwak, P. Rodriguez, Y.-Y. Ahn, and S. Moon. I tube, you tube, everybody tubes: Analyzing the world's largest user generated content video system. In *Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement* (*IMC'07*), pp. 1–14, San Diego, CA, October 2007.
9. S. Chandra. Content adaptation and transcoding. In *Practical Handbook of Internet Computing*, Munindar P. Singh (ed.), pp. 1–144, Chapman & Hall/CRC Press, Baton Rouge, LA, 2004.
10. C.-Y. Chang and M.-S. Chen. On exploring aggregate effect for efficient cache replacement in transcoding proxies. *IEEE Transactions on Parallel and Distributed Systems*, 14:611–624, June 2003.

11. H.-H. Chen, M. Guizani, and W. Mohr. Evolution toward 4G wireless networking. *IEEE Network*, 21(1):4–5, January/February 2007.
12. Y.-K. Chen and S.-Y. Kung. Trends and challenges with system-on-chip technology for multimedia system design. In *Proceedings of Emerging Information Technology Conference*, p. 4, Santa Clara, CA, March 2005.
13. G. Cormode and B. Krishnamurthy. Key differences between Web 1.0 and Web 2.0. *First Monday*, 13(6), June 2008.
14. M. Eiriniaki and M. Vazirgiannis. Web mining for web personalization. *ACM Transaction on Internet Technology*, 3(1):1–27, 2003.
15. S. Flesca, S. Greco, A. Tagarelli, and E. Zumpano. Mining user preferences, page content and usage to personalize website navigation. *World Wide Web*, 8(3):317–345, August 2005.
16. L. Guo, E. Tan, S. Chen, Z. Xiao, O. Spatscheck, and X. Zhang. Delving into Internet streaming media delivery: A quality and resource utilization perspective. In *Proceedings of the 6th ACM SIGCOMM Conference on Internet Measurement* (*IMC'06*), pp. 217–230, ACM, New York, 2006.
17. A. Jaokar and T. Fish. *MobileWeb 2.0: The Innovator's Guide to Developing and Marketing Next Generation Wireless/Mobile Applications*. Futuretext, London, U.K., 2006.
18. T.-P. Liang, H.-J. Lai, and Y.-C. Ku. Personalized content recommendation and user satisfaction: Theoretical synthesis and empirical findings. *Journal of Management Information Systems*, 23(3):45–70, 2007.
19. Market Intelligence Center. Global mobile phone subscribers forecasted to reach 4.5 billion by 2012. Press Release, 2008.
AQ8 20. T. O'Reilly. What Is Web 2.0? O'Reilly, 2005.
21. M. Pati ~no-Mart′inez, R. Jim'enez-Peris, B. Kemme, and G. Alonso. Consistent database replication at the middleware level. *ACM Transactions on Computer Systems*, 23(4):1–49, 2005.
22. P. Research. 950 million users will access social networking sites via mobile devices. February 2008.
23. H. Schwarz, D. Marpe, and T. Wiegand. Overview of the scalable video coding extension of the H.264/AVC standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 17(9):1103–1120, 2007.
24. N. Serrano and J. P. Aroztegi. Ajax frameworks in interactive web apps. *IEEE Software*, 24(5):12–14, September–October 2007.
25. W. Shi, K. Shah, Y. Mao, and V. Chaudhary. Tuxedo: A peer-to-peer caching system. In *Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications* (*PDPTA'03*), pp. 981–987, Las Vegas, NV, June 2003.
26. W. Shu, M. Jungwon, and K. Y. Byung. Location based services for mobiles: Technologies and standards. In *Proceedings of the IEEE International Conference on Communication* (*ICC'08*), Beijing, China, May 2008.
27. G. Singh. Guest editor's introduction: Content repurposing. *IEEE Multimedia*, 11(1):20–21, March 2004.
28. S. R. Subramanya and B. K. Yi. Enhancing the user experience in mobile phones. *Computer*, 40(12):114–117, 2007.
29. Y. Sun, T. L. Porta, and P. Kermani. A flexible privacy-enhanced location-based services system framework and practice. *IEEE Transactions on Mobile Computing*, 8(3):304–321, March 2009.
30. TechCrunch. YouTube Mobile Uploads Up 400Release. June 2009.

31. S. J. Vaughan-Nichols. Will mobile computing's future be location, location, location? *IEEE Computer Magazine*, 42(22):14–17, 2009.
32. M. Walker, R. Turnbull, and N. Sim. Future mobile devices? An overview of emerging device trends, and the impact on future converged services. *BT Technology Journal*, 25(2):120 – 125, April 2007.
33. T. Yamakami. A Zipf-like distribution of popularity and hits in the mobile web pages with short life time. In *Proceedings of the International Conference on Parallel and Distributed Computing, Applications and Technologies* (*PDCAT'06*), pp. 240–243, Taipei, Taiwan, December 2006.
34. D. Zhang. Web content adaptation for mobile handheld devices. *Communications of the ACM*, 50(2):75–79, 2007.

Author Queries

[AQ1] Please provide caption for Figure 5.1.
[AQ2] Please check if the edits to the sentence starting: "The upload… POI discovery services." are ok.
[AQ3] Please check if the edits to the sentence starting: "Although… is possible." are ok.
[AQ4] Please check if the inserted expansion for the acronym API is ok.
[AQ5] Please check if the inserted expansion for the acronym SDK is ok.
[AQ6] Please check if the inserted expansion for the acronym MMS is ok.
[AQ7] Please check if the inserted expansion for the acronym RSS is ok.
[AQ8] Please provide complete details in Refs. 22 and 30, if appropriate.