A Distributed Architecture for Management and Retrieval of Extended Points of Interest[*]

Claudio Bettini
DICo - University of Milan, Italy
bettini@dico.unimi.it

Nicolò Cesa-Bianchi
DSI - University of Milan, Italy
cesa-bianchi@dsi.unimi.it

Daniele Riboni
DICo - University of Milan, Italy
riboni@dico.unimi.it

Abstract

This paper presents a distributed architecture for the management and retrieval of particular objects called POIsmarts. POIsmarts can be considered as the convergence between virtual points of interest (web bookmarks) and physical points of interest (gps bookmarks). We provide a XML schema defining the structure and properties of POIsmarts and we show how these objects can be stored, shared, searched and accessed in a uniform way independently from the user device. The paper also reports preliminary results on performing automatic categorization and search (in personal or shared POIsmart hierarchies) using machine learning techniques.

1 Introduction

There is a rapidly growing number of users of GPS enabled mobile devices, and this or a similar technology for accurate localization will be eventually integrated in most mobile phones. The notion of *Point of Interest* used by navigation software to trace and highlight resources possibly interesting to the user is analogous to the one of a web page bookmark, except that coordinates are used to identify the resource instead of a URL. It is also likely that some resources have both a web site and a physical location. Consider, for example, a university department, a restaurant, or a museum. We call POIsmart the object used to describe the geographical as well as the virtual location of a resource. We propose a XML Schema to define the POIsmart data structure; on one side the schema guarantees backward compatibility to all the bookmark formats used by major browsers as well as to the formats of points of interest used by major navigation software. This means that we can import and export data from and to these applications. On the other side, the schema integrates these informations and supports multimedia annotation of resources as well as special fields for advanced search and sharing.

Our research project is not just focused on the definition of this data structure, but aims at defining an advanced architecture for managing, sharing and searching POIsmarts. An extension of current web browser bookmarking facilities to deal with POIsmarts would be highly unsatisfactory since they already have a number of recognized drawbacks, especially for experienced users managing a large and complex bookmark hierarchy, and accessing their bookmarks from different devices. On the other side, current navigation software has very primitive POI management facilities if any.

The main features we envision for our POIsmart management system are: (1) a POIsmart server enabling each user to access his POIsmarts from different browsers and different devices (desktop, PDA, cellular phone, ...); (2) a facility to share POIsmarts with other users; (3) an adaptive categorization system, suggesting appropriate folders upon user bookmarking of specific POIsmarts; (4) a search facility to quickly find private and/or shared POIsmarts based on context data and free text search terms.

The architecture presented in the paper is integrated with a middleware for the management of context data, including user profiles, device capabilities, network status and service policies [1].

A number of shareware utilities has been developed for web bookmarks mainly addressing 1 and 3, and prototypes have been developed to address 2 and 4 (see e.g. [7]), but none of them integrates nicely these features, and, to our knowledge, an extension to manage physical locations was never considered. On the other side, the idea of bookmarking physical locations is not new [3], and in the last years it

[*]This work has been partially supported by Italian MIUR (FIRB "Web-Minds" project N. RBNE01WEJT_005).

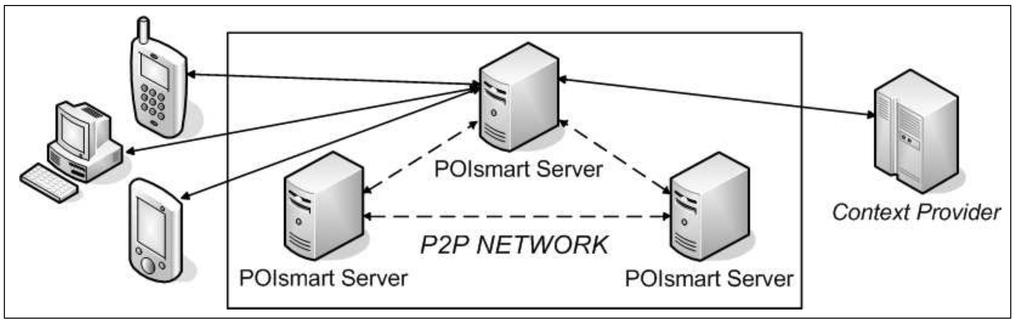

Figure 1. The architecture for POIsmart management

has been exploited for different means in the fields of ubiquitous and pervasive computing. For example, [9] and [2] present architectures for managing virtual notes that allow the user to attach comments, reminders, and multimedia resources to objects identified by a physical location. A key difference with our work is that these systems are mainly intended for personal use, while in our proposal the sharing of POIsmarts between (communities of) users is one of the main goals. The introduction of sharing poses additional issues involving the management and search of resources.

The paper is structured as follows: In Section 2 we show the main components of the architecture, and some possible variants. In Section 3 we concentrate on the problem of automatic categorization and search of POIsmarts. In Section 4 we report the current implementation of our system, and in Section 5 we discuss future work and summarize our conclusions.

2 Architecture

The system architecture we propose, sketched in Figure 1, is composed of two levels:

- POIsmart Servers (PS) for the management of the device-independent representation of POIsmarts;
- Client Systems (CS) as device-dependent interfaces for the client-side management and search of POIsmarts.

2.1 POIsmart Servers

The main component of the architecture is the POIsmart Server (PS) which manages the representation of POIsmarts for one or more users. There are three types of requests received by each PS from a client system: (a) requests for specific folders identified by their name in the POIsmart hierarchy; (b) queries based on search terms and context data; (c) requests to add, delete or modify POIsmart and/or folders. The PS answers to requests of type (a) and (b) by providing folders in the format appropriate for the specific browser and device. Requests of type (b) are answered using a multi-feature query engine and machine learning techniques (see Section 3). Similar techniques are used to handle requests to insert new POIsmarts by suggesting a list of candidate folders where to store the POIsmarts.

In order to support POIsmart sharing, PSs are organized as nodes in a peer-to-peer network. Requests of type (b) are forwarded from the local PS to all of its neighboring nodes. Each of these nodes evaluates the request and in turn forwards it to its neighboring nodes. Typically, a PS can reside on a department server, an ISP, or even on a personal server. The peer-to-peer PS network organization will also allow a client system to connect to an arbitrary PS in the network and to transfer and operate on its POIsmarts through it.

Figure 2 shows the data flow upon a user's request. In Step 1, the user submits a query based on zero or more keywords. The POIsmart Server queries the Context Provider for the user's profile information and other context data (Step 2). This information as well as the original query is used by the *Query creation* module to build a multi-feature query (Step 3), which also specifies the maximum number

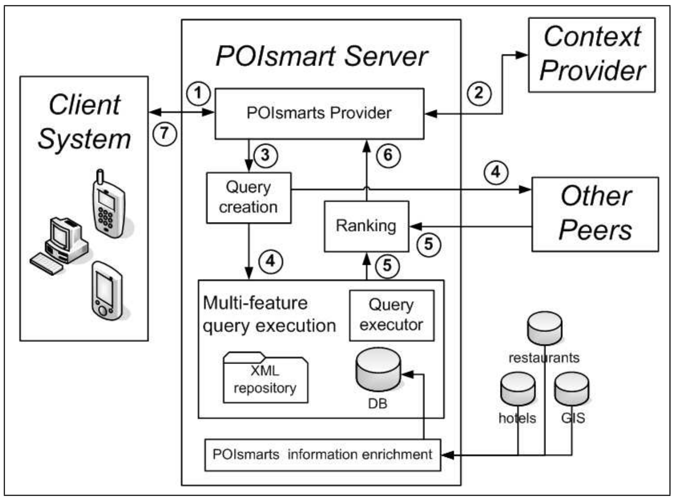

Figure 2. The POIsmart Server

(n) of POIsmarts to be returned.[1] Next (Step 4), the query is forwarded to the known peers, and executed locally by the *Multi-feature query execution* module. In Step 5, each known peer returns the XML representations of the top n POIsmarts, each one with the associated confidence value. The returned POIsmarts, together with the ones retrieved locally, are provided to the *Ranking* module. Finally, this module selects the top n POIsmarts, which are sent to the user's device (Step 6 and 7).

When a peer receives a query from another peer in the network, it performs the same operations described above, except that the multi-feature query is already available and that it will forward the query only to a subset of the peers. Moreover, the *Ranking* module does not return its result to the client but to the peer that made the request.

2.2 Client Systems

Client Systems (CSs) provide an interface for the user to select POIsmarts, add new POIsmarts (including automatic categorization by contacting a *reference* PS), and reorganize POIsmarts. They also provide a way to search private POIsmarts and other users' shared folders. For efficiency and privacy reasons, each CS is configured to connect to a reference PS that stores its POIsmart hierarchy.

CSs are device-dependent, since the graphical interface and several features need to adapt differently to desktop computers, PDAs, cellular phones and other devices. Different features are implemented depending on the device capabilities. For example, the CS on devices that have enough memory and that are not always online work using a local copy of the hierarchy and synchronize with the PS upon user request; on other devices only part of the POIsmart hierarchy is transferred to the client, and any change to this transferred part is immediately propagated to the PS.

2.3 The XML Schema for POIsmarts

In order to simplify the exchange of POIsmarts between different software architectures (e.g., from a Java PS to a .Net smart client), POIsmarts are represented in XML. The POIsmart XML Schema is derived in part from the Document Type Definition proposed for the XML Bookmark Exchange Language (XBEL) [5], an interchange format for hierarchical bookmark data supported, among others, by the Galeon browser and the Konqueror file manager. Our schema defines a superset of the information used by bookmark managers integrated in popular browsers and of the information used by managers of points of interest. Each POIsmart in the schema represents a resource by describing the real-word and/or virtual features of the resource (e.g.,

[1]The parameter n is chosen based on the device capabilities and the available connectivity.

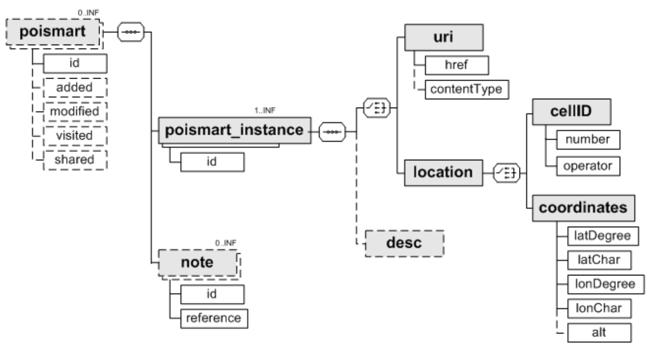

Figure 3. A fragment of the XML Schema defining POIsmarts

its physical location, its description, and URIs of web content related to it). Figure 3 provides a diagram of part of the Schema. These are the main characteristics of the schema:

- Each *poismart* entry contains zero or more multimedia *notes* and *poismart_instances*.

- Each note contains a reference to a multimedia web resource related to the POIsmart (e.g., a video describing the resource, or a vocal comment on the resource recorded by a user).

- The notion of POIsmart instance has been introduced to model the common case of a resource, like a department store chain, that can have multiple branches, each one with a different location and possibly a different web site. Each POIsmart instance can contain a URI and/or location information: while the former identifies a web site related to the instance, the latter provides its physical location (which can be either a GPS coordinate or a cell ID). It is worth to note that a single POIsmart instance can contain more URIs: this is useful when the site offers different entry points for different technologies (XHTML, WML, etc.).

- The schema defines appropriate tags to support the automatic categorization of POIsmarts and folders. The *metadata* tag, in addition to text keywords, contains an element *index* that is used for storing the *feature vector* of the web pages referenced by the POIsmart. This attribute is used both for the suggestions on the most appropriate folders for a new POIsmart and for the search of desired POIsmarts or folders.

- Finally, POIsmarts and folders have a *owner* attribute and a boolean *shared* attribute; the latter states whether they are accessible only by their owner or not.

2.4 Peer-to-peer architecture

The choice of structuring POIsmart Servers as nodes in a peer-to-peer network is mainly motivated by the need of providing ubiquitous access and research capabilities to the users of the system. As explained in section 2.1, users' queries are forwarded across the network for finding interesting items in the set of POIsmarts shared by other people. Moreover, there are several reasons, e.g., the user temporary location, why a CS may occasionally connect to a PS which is not its reference PS. Additionally, peer-to-peer seems to be the best choice for addressing the problem of scalability in decentralized communities.

Our peer-to-peer infrastructure essentially follows the model of the well-known Gnutella protocol, but attempts to overcome its main weaknesses by improving message routing. The mechanism for discovering other peers on the network works as follows: When a new peer (PS) needs to join the peer-to-peer network of PSs, it needs the address of a known PS that is used as a *Host Cache Service*. The new PS sends a request to this host obtaining a (possibly partial) list of peers participating in the network. Each peer in this list can also be contacted to retrieve more peer addresses. Moreover, new peers are added to the set of known ones in the search phase. For searching POIsmarts or folders in the network, starting from a specific peer p_1, we forward a *Query* message to every peer known to p_1. When a peer p_i forwards a query to p_j, it provides it with the set of its known peers. When p_j receives the set of peers known by p_i, it adds them to its set of known peers. Thus, p_j forwards the query only to peers known by itself which are unknown to p_i. Note that the set of peers known by p_j is a superset of the set of peers known by p_i. Every message has a TTL field that states the maximum number of times the message can be forwarded. Moreover, when a peer receives for the second time the same message, it does not forward it to any other peer. However, since no assumption can be made with regard to the storing of POIsmarts in specific sets of peers, search of POIsmarts is essentially performed by flooding the whole network of peers with query messages (until the TTL expires). Obviously, this approach is far from optimal, since every peer should be contacted in order to correctly answer a query.

As a possible alternative, we are investigating the possibility to take into account the location of both POIsmarts and peers, in order to assign POIsmarts to "close" peers. This feature would allow to answer location-based queries only by contacting those peers that cover the area of the query. However, the issue of implementing this feature preserving load-balancing is particularly challenging, since generally the distribution of peers and POIsmarts is different (i.e., a particular area can have a lot of points of interest but a small number of peers, and vice versa), and is left to future work.

3 Classification and Search

Assigning new POIsmarts to folders in a large and complex hierarchy may be a tedious task, specially when using mobile devices. This motivates the introduction in our architecture of an automatic categorization system, which suggests the most appropriate folders for each new POIsmart.

3.1 New POIsmart categorization

Each time a user wants to save a new POIsmart, the PS suggests an ordered list of folders chosen from the current folder hierarchy. The suggestion system exploits both the virtual and the physical features of the POIsmart.

When the POIsmart contains one or more references to web sites, the web sites content is used to categorize the POIsmart using machine learning techniques. The system maintains a local dictionary of words that occurred so far in web pages referenced by the POIsmarts of the user. The dictionary is used to represent a web page as a *page feature vector* using the standard vector space model of information retrieval [8]: roughly speaking, each vector coordinate measures the contribution of the corresponding dictionary word in the semantics of the page. The categorization system also keeps, for each folder i in the hierarchy, a *folder feature vector* w_i. Given a new POIsmart to add, the system computes the page feature vectors $v_1, \ldots, v_n$ of the n web pages referenced by the POIsmart and then suggests to the user the name of one or more folders in the hierarchy. A folder i is suggested for labeling a web page v_j (and, thus, the POIsmart which contains a reference to it) whenever the confidence for that folder, computed as the cosine of the angle between w_i and v_j, goes above a certain threshold. Then the user can choose whether to assign the POIsmart to one or more folders in the set suggested by the system, or to assign it to another folder. Note that, unlike other web page categorization systems, in order to provide real-time responses our system does not use any information taken from pages linked to the target pages.

In the case the new POIsmart does not contain references to web sites, the categorization technique exploits the physical location of the resource and the user's context information. The POIsmart location is used by a GIS to retrieve a set of points of interest close to it. The number of points of interest in the set generally depends on the precision of the localization technology. For instance, the area identified by a cell ID can be wide, and would possibly contain a considerable number of points of interest. The GPS system provides very accurate location information, but typically it does not work indoor, like in shopping centers or museums;

thus, the location area, derived from the last available GPS data, must be widened in order to ensure that the right point of interest is included. The folders containing the points of interest in the set become candidates for including the new POIsmart and they are ranked taking into account the user's context.

Example 1 *Consider the case of John, a user who had just had dinner in a restaurant. Being very satisfied, he decides to add a POIsmart for the restaurant with a short vocal message in order to remember the experience and to share it with his friends. Since the GPS signal was absent in the restaurant, the available coordinates are the ones corresponding to the parking lot in front of the restaurant. The PS, in order to suggest a folder where to store the new POIsmart, queries a GIS sending the coordinates and asking for points of interest in a range of 50 meters. The GIS returns 4 points of interest corresponding to the parking lot, the restaurant, a flowers shop, and a night club. The PS retrieves context data from the Context Provider, among which the current time. Since the current time is 1p.m., the system ranks very low the categories of "night clubs" and "specialty shops" (since most are closed at that time), and it suggests "Restaurants" as the probable category for the POIsmart, followed by "Parking lots".*

3.2 POIsmarts retrieval

A Client System can query its reference PS using free text search terms. The query is transformed by a proper module on the PS into a multi-feature query by considering context information, such as his current location, device capabilities, known interests, and possibly his current activity.

To answer such a multi-feature query, the PS adopts a ranking algorithm similar to the one proposed in [6]. For each considered feature, a single degree of match is computed. For instance, the degree of match of the terms in the search expression against the set of available POIsmarts is obtained computing a *query feature vector* q based on the search terms, and then calculating the cosine of the angle between q and the feature vector w of the POIsmart. The single degrees of match are then combined to obtain a comprehensive confidence value.

3.3 Experimental results

Our experimental results on automatic classification are currently limited to POIsmarts having URI information since we built feature vectors only based on the content of the corresponding web pages. Previous experimental results in web page categorization [10] show that term-weighting schemes (taking into account the semistructured nature of HTML) and dimensionality reduction techniques

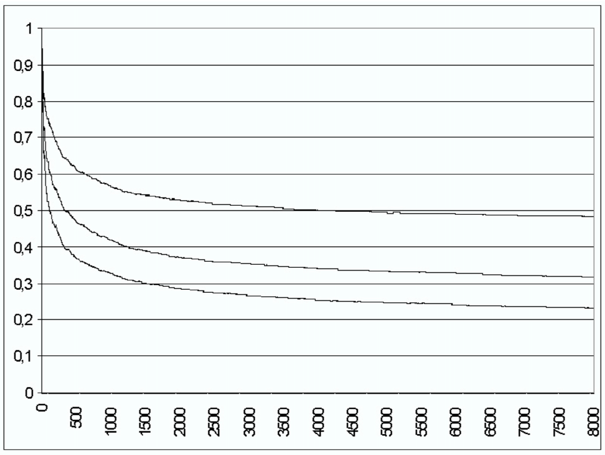

Figure 4. Preliminary results for the categorization system. Curves display error rates related to finding the correct folder within the first three (lower curve), two (middle curve), or within the first folder (upper curve) in the ranking generated by the categorizer.

can both improve the classification performance. These techniques have a low computational cost and are therefore suitable for our real-time application. In particular, term-weighting techniques exploit the structural information present in HTML documents by considering not only the number of occurrences of terms in documents, but also the HTML element the terms belong to. The experimental results show that varying the weights of a term depending on the associated HTML element, e.g. assigning greater importance to terms that belong to the `META` and `TITLE` elements, may lead to an improvement in accuracy. This weighting method, called *structure-oriented weighting technique (SWT)* in [10], is defined by the function

$$\mathrm{SWT}_w(t_i, d_j) = \sum_{e_k} \Big(w(e_k) \cdot \mathrm{TF}(t_i, e_k, d_j) \Big)$$

where e_k is an HTML element, $w(e_k)$ denotes the weight we assign to the element e_k and $\mathrm{TF}(t_i, e_k, d_j)$ denotes the number of times the term t_i is present in the element e_k of the HTML document d_j.

We performed various experiments on a corpus of 8000 web pages belonging to 10 Yahoo! categories. Page feature vectors are built by applying the *structure-oriented weighting technique* together with a strong feature selection. To compute the folder feature vectors w_i we use the perceptron, an extremely fast and incremental algorithm. The perceptron adjusts the vectors w_i after each insertion of a new POIsmart containing references to web pages in the hierarchy, thus improving its performance as the hierarchy grows.

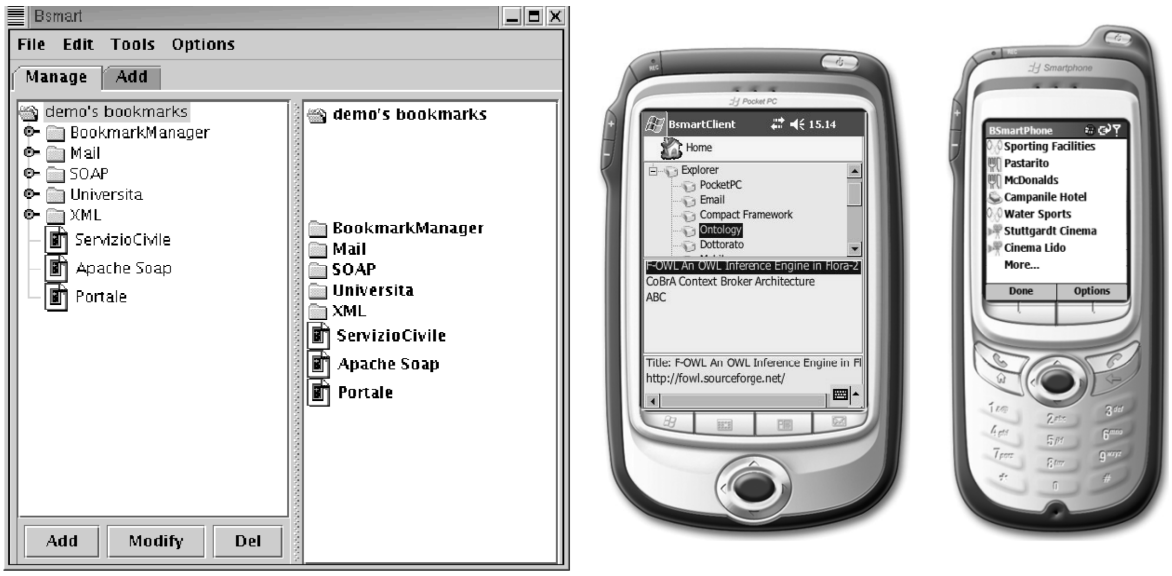

Figure 5. Client systems

For each new web page taken from the corpus, we ranked the 10 folders according to the cosine of the angle between the page feature vector and each folder feature vector. An error occurs when the right category is not in the first k places of the ranking, where we set $k = 1, 2, 3$. Figure 4 shows the corresponding error rates against the number of pages. Though these results are very preliminary and not yet satisfying in terms of performance, note that the error rates have a nice shape; i.e., they drop quickly at the beginning of the learning process. To decrease the error rate further we plan to use variants of the perceptron tailored to ranking problems, such as the one proposed in [4].

4 The current system prototype

A POIsmart management system prototype is being developed at the University of Milan with the goal of testing the various components of the proposed architecture and eventually obtain a fully functional POIsmarts management system. The prototype currently implements part of the features explained in the previous sections.

Our architecture adopts web services for client/server and server/server communication. The choice of web services is mainly motivated by the need for device independence: users should be able to access their POIsmarts management system from many different devices. Many of these devices need an ad hoc client software for the communication with the reference PS. We currently consider web services to be the best and simplest solution for the interaction of applications running on different environments, compared to proprietary technologies like DCOM or RMI.

Java-based POIsmart servers can currently manage multiple users XML POIsmart representations and can interact with client systems allowing insertion and deletion of POIsmarts and folders. The peer-to-peer algorithm is also implemented and working as described in Section 2.4. We are still working on the integration with the middleware providing context information, and hence multi-feature queries are not yet supported.

POIsmart servers currently support basic searches on private as well as on shared POIsmarts hierarchies. The search facility we implemented is based on keyword matching and location; hence, the list returned by the server includes only POIsmarts such that these keywords are found in its title or metadata keyword tag, ordered by their proximity to the user. The search facility allows the user to specify if the search should only span over the personal hierarchy or should extend to other users shared POIsmarts, both on the same PS and on different ones in the peer-to-peer network.

One of the main goals of our project is to allow users to access their POIsmarts from different devices. Currently, we have developed Client Systems for the most commonly used devices, i.e. personal computers, cellular phones and PDAs (see Figure 5). The client for personal computers is a multi-platform Java standalone application that provides a user-friendly interface for the interaction between the end user and the system. When the user connects to his PS, he obtains his whole POIsmarts hierarchy. The user can manipulate it by adding, modifying or deleting POIsmarts and POIsmart folders. When the user wants to save his changes, he makes a request to his reference PS, sending an appropriate XML representation of the changes made by the user in the current session. The user can also search for POIsmarts or POIsmart folders shared by other users. The clients for the PDA and cellular phone platforms have been developed using the Java 2 Microedition platform. This choice guarantees the reuse of certain Java components, and provides adequate robustness and portability. We have a version for cellular phones supporting MIDP 1.0 and one for PDAs implementing the J2ME Personal Profile specification. The communication between the client and PSs is realized by using kSOAP, an implementation of a reduced set of the SOAP protocol specifications, suitable for the Java 2 Microedition. In addition to Java smart clients, we developed a .Net version of the client for Microsoft Smartphone 2003 devices. This solution seems to be more appropriate for mobile phones equipped with the .Net Compact Framework. The clients for mobile devices essentially have the same functionality of the personal computer version. The key differences are due to the limited amount of memory available on mobile devices. For example, in the MIDP 1.0 version no POIsmart is stored on the device and all operations are carried out server-side, while in the version for PDA, the POIsmart hierarchy can be cached locally and synchronized with the server with an appropriate policy.

The capability of the client systems to automatically launch a browser when selecting a URL as well as the capability of a browser to store a URL in the POIsmart hierarchy are key factors for the usability of the proposed system. For this reason, we are working for obtaining a full integration with the most widely used browsers. We have managed to include the former capability in all versions of the clients except the MIDP 1.0 one. Regarding the latter capability, the integration has been done between Internet Explorer and the client for personal computers: a specific toolbar button has been added on the browser which users can use to create a new POIsmart which has a reference to the web page currently displayed. The integration is more complex for microbrowsers which usually are less customizable; we are evaluating several solutions one of which consists in the association of a hardware button to the creation of a new POIsmart.

5 Conclusions and future work

In this paper we introduced the new notion of POIsmarts as the convergence between virtual points of interest (web bookmarks) and physical points of interest (gps bookmarks) and proposed an architecture for advanced management and retrieval of this type of data. The key advantages of a system based on this architecture are the possibility for each user to access his POIsmarts from every device and location, and the possibility to obtain new POIsmarts from people that share common interests. We have a running prototype of the main modules of the architecture and client systems for PCs as well as for mobile devices. We plan to use access control policies and other mechanisms to address the users privacy concerns about sharing POIsmarts. However, a detailed description of privacy policies is out of scope for this paper.

In addition to technical advances in the implementation of client systems and in their interfacing with browsers and navigation software, we are currently working on the integration with the middleware infrastructure for context data provisioning illustrated in [1], and on the definition of a peer-to-peer protocol that takes into account the location of both peers and POIsmarts.

Acknowledgements

We would like to thank Carlo Cestana, Alessandro Lunadei, and Marco Ronchi for implementing parts of the prototype, and Marko Grobelnik for providing the classified web pages we used for our experiments.

References

[1] 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)*, pages 121–134. Springer, 2004.

[2] G. Borriello, W. Brunette, M. Hall, C. Hartung, and C. Tangney. Reminding About Tagged Objects Using Passive RFIDs. In *Ubicomp*, pages 36–53. Springer, 2004.

[3] P. J. Brown. The Electronic Post-it Note: a Metaphor for Mobile Computing Applications. In *IEE Colloquium on Mobile Computing and its Applications*, 1995.

[4] K. Crammer and Y. Singer. A New Family of Online Algorithms for Category Ranking. In *SIGIR 2002 Proceedings*, 2002.

[5] J. Fred L. Drake. The XML Bookmark Exchange Language. Technical report, Corporation for National Research Initiatives (CNRI), USA.

[6] U. Güntzer, W.-T. Balke, and W. Kießling. Optimizing Multi-Feature Queries for Image Databases. In *VLDB 2000, Proceedings of 26th International Conference on Very Large Data Bases*, pages 419–428, 2000.

[7] W. Li, Q. Vu, D. Agrawal, Y. Hara, and H. Takano. Powerbookmarks: a System for Personalizable Web Information Organization, Sharing, and Management. *Computer Networks*, 31(11):1375–1389, 1999.

[8] C. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, 1999.

[9] J. Pascoe. The Stick-e Note Architecture: Extending the Interface Beyond the User. In *IUI '97: Proceedings of the 2nd international conference on Intelligent user interfaces*, pages 261–264. ACM Press, 1997.

[10] D. Riboni. Feature Selection for Web Page Classification. In *EURASIA-ICT 2002 Proceedings of the Workshop*, pages 473–477, 2002.