The W4 Model and Infrastructure for Context-aware Browsing The World

Gabriella Castelli, Alberto Rosi Marco Mamei, Franco Zambonelli

***Abstract*—The imminent mass deployment of pervasive computing technologies such as sensor networks and RFID tags, together with the increasing participation of the Web community in feeding geo-located information within tools such as Google Earth, will soon make available an incredible amount of information about the physical and social worlds and their processes. This opens up the possibility of exploiting all such information for the provisioning of pervasive context-aware services for "browsing the world", i.e., for facilitating users in gathering information about the world, interacting with it, and understanding it. However, for this to occur, proper models and infrastructures must be developed. In this paper we propose a simple model for the representation of contextual information, the design and implementation of a general infrastructure for browsing the world, as well as some exemplar services we have implemented over it.**

***Index Terms*—Pervasive computing, Browsing the world, GIS, GPS, RFID tags.**

I. Introduction

Two apparently disjoint trends motivate this work. On the one hand, the imminent mass diffusion of pervasive computing technologies such as sensor networks [ChoK03] and RFID tags [Wan06] will soon make available an incredible amount of real-time information about the physical world, its processes, and its objects. On the other hand, the dramatic success of participatory Web tools (aka Web 2.0 technologies) is feeding the Web with information of any kind about any topic. In particular, mapping tools such as Google Earth and Google Maps get continuously enriched by geo-located information coming from very diverse social communities and related to a variety of facts and events situated in the world [But06].

Overall, both the above trends contribute to accumulate information that can be potentially used to build real-time and historical models of a number of facts and processes happening in the world. More pragmatically, the possibility of acquiring detailed digital information about the surrounding context opens up the possibility of exploiting all such information for "*browsing the world*" [Cas06]. The concept of browsing the world considers that, by properly integrating information about the surrounding world coming from both pervasive devices and form the Web, it will be possible for users to gather contextualized relevant information, and for services to effectively support user activities related to interacting with the physical world in a context-aware way.

However, considering that the amount of available information from a variety of sources could become overwhelming, its effective exploitation by users and services calls for proper models to represent such data in an expressive yet simple-to-be-manipulated way, and for proper software infrastructure to organize and provide access to it. Accordingly, the contribution of this paper is twofold.

First, we propose a simple model to represent contextual information about the physical world, for the use of both users' querying activities and context-aware services. The model, which we call "W4", is based on the consideration that most information about the world can be simply represented in terms of four "W"s – *Who*, *What*, *Where*, *When* – and that such a representation enables for very expressive, and flexible data usages.

Second, we describe the design and implementation of a general middleware infrastructure for browsing the world, facilitating the development and supporting the activities of general-purpose context-aware pervasive services. The infrastructure supports PDAs and laptops access to information coming from both pervasive devices and the Web, provides for representation and organization of data in W4 terms, makes available a Java interface for users' queries and for services access to such data, and it is integrated with both Google Earth and Google Map for the sake of effective user interfacing.

The remainder of this paper is organized as follows. Section 2 better details the general scenario of browsing the world and the challenges it implies. Section 3 presents the W4 model. Section 4 details the implemented software infrastructure. Section 5 presents some services we have implemented on top of our system. Section 6 discusses related work in the area. Section 7 concludes.

II. Browsing the World

In this section, we better define the scenario in which our research situates, by properly identifying the components involved in the "browsing the world" vision, and by discussing the associated key challenges.

A. Scenarios

As stated in the introduction, in the near future, our everyday environments will be densely populated by a variety

Manuscript received July 7, 2006. This work was supported in part by the project CASCADAS (IST-027807) funded by the FET Program of the European Commission.

Gabriella Castelli, Alberto Rosi, Marco Mamei, Franco Zambonelli are with the Dipartimento di Scienze e Metodi dell'Ingegneria. Università di Modena e Reggio Emilia, Italy; e-mail: name.surname@unimore.it.

of embedded devices such as sensor networks [ChoK03], RFID tags [Wan06]. Users in an environment will be able, via wireless interfaces mounted on some wearable computing device (e.g. a PDA or a smart phone), to directly access devices in their proximities to gather information about phenomena occurring in the surroundings or (as in the case of RFID tags attached to objects) about nearby physical objects. In addition, users will be able to access to the Web via some wireless communication technology, to dynamically retrieve any needed information. Other than accessing "traditional" Web information (e.g., html pages and Web services), this also enables users to access geo-located information concerning specific sites geographical areas and general facts and annotations about them, as they can continuously provided via collaborative Web 2.0 technologies by the Web community [Esp01, TerK06]. In addition, it enables users to access information generated by sensors and embedded devices (far in the world or close to him but beside his range of direct access).

Users, in turn, can decide to unveil (totally or to some limited extent) their presence in an environment, by making somehow available to the public their identity, location, and/or activities. This can occur by dynamically uploading such information on the Web, or by making it available to other via ad-hoc connections, or even by uploading it into surrounding pervasive devices. In this latter case, pervasive devices such as RFID tags would act as a sort distributed memory infrastructure [MamQZ06]. The location of users will be always available, either because they will carry on a GPS or because of location can be inferred by the patterns of access to pervasive devices (e.g., the access to a RFID tag with a known location implicitly determines the location of the user [Sat05]) (see Fig. 1).

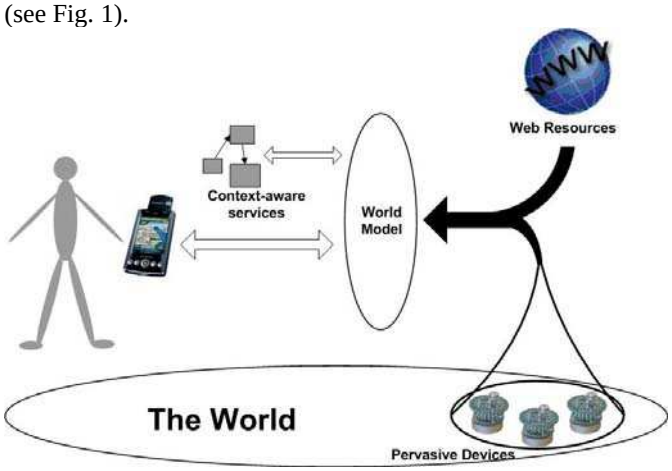

Figure 1. The general scenario of browsing the world.

On the basis of the above considerations, the concept of browsing the world, in general terms, consider the possibility of navigating in an information space that – by properly merging and integrating information coming from both pervasive devices and the Web can represent a detailed model of the world, comprising both present and historic fine-grained geo-located data about the world, its entities, its processes, and its social life. In context-aware user-centric terms, which are the ones of more interest here, the concept of browsing the world implies the possibility for users in an environment to access and navigate meaningful information about the surrounding physical world, and for software services to access and manipulate such information to enforce various degree of context-awareness and context-adaptation.

B. Challenges

From the merely technological viewpoint, the "browsing the world" vision could be already turned into reality. Indeed, novel services and Web sites that can be included in the "browsing the world" category appear every day [Cas06]. However, beside specific service implementations, for browsing the world to become common practice based on sound engineered activities, several challenges remains to be addressed. In particular:

1. It would be fundamental to create a general model to represent context information and to build a world model. Spatial information is important but it is not enough. Temporal information must be included, as well as information describing the activities taking place in the world. The model should enable to deal with incomplete information, and should allow navigation among context information on the basis of what is available at a given time.
2. It would be important to have a general infrastructure supporting the model that should work without requiring or committing to the availability of specific technologies. The infrastructure should be general-purpose, autonomic and adaptable. Relying on this infrastructure, the activities of browsing the world should not be compromised because of say, the temporal unavailability of an Internet connection or the unavailability of a GPS, or of an RFID reader. Consequently, applications built on that infrastructure should not mandate the availability of specific information, but should exploit whatever available information on a best effort basis.

Beyond the horizon, it would be important for such a general model to enable easy processing of data, to facilitate the identification of links between isolated bunch of information. This would enable the creation of complex knowledge networks, and possibly would promote the creation of "new knowledge", as it can be derived by inference from existing information [Bau06].

The attempt to face the above challenges, by defining a simple yet effective model for context data and a general software infrastructure, as preliminary and incomplete as it can be, is the exact goal of our work.

III. THE W4 CONTEXT MODEL

We propose a simple model in which context data is expressed by a four field structure: who, what, where and when. Such a model appears effective in a number of circumstances since it points out some of the main topics that are also involved in human thinking: who is acting? What is he/she/it doing? Where and when the action takes place?

A. Overview

The goal of our proposal is to develop a general model to manage contextual information. Information to be handled will come from multiple and heterogeneous sources, and would be related to a large number of situations ranging from the description of physical properties in geographic areas, to social facts and processes happening in the world.

In particular, we developed a model in which context data is described by means of 4-fields tuple: (Who, What, Where, When). We chose this structure because of its evident meaning and flexibility. In fact, a W4-tuple allows to express a situation in a rather natural and human-like way, e.g., "someone or something (*Who*) does some activity (*What*) in a certain place (*Where*) at a specific time (*When*)". We call each of these tuples a *knowledge atom* to describe the fact they represent an atomic unit of context information.

Knowledge atoms are created by a number of software agents running on different (possibly embedded) devices, and will be stored in a suitable shared data space (in section 4, we detail our actual implementation of this space). Application agents that can range from context-aware service providers, to simple interfaces supporting users in browsing information, will access the shared space to retrieve those context information that are suitable for their application task.

B. W4 Data Representation and Generation

We define context as a four-field tuple (*Who*, *What*, *Where*, *When*):

- ***Who*** is the subject described by the context structure. *Who* may be a human person (e.g., Gabriella) or an unanimated part of the context (e.g., an RFID tag). The *Who* field is represented by a string with an associated namespace that defines the "kind" of entity that is represented. For example, valid entries for this field are: "person:Gabriella", "tag:tag#567".
- ***What*** is the activity performed by the subject. In the likely case that this is not directly available, it can be inferred from the other context parameters (e.g., an accelerometer can reveal that the user is jogging), or it ca be explicitly supplied by the user. This field is represented as a string containing a predicate-complement statement. For example, valid entries for the *What* field are: "read:book", "work:pervasive computing group", "read:temperature=23".
- ***Where*** is the location to which the context relates. In our model the location may be a physical space represented by coordinates (longitude, latitude) or by geographic regions (specifically, our model adopts the Postgis language to describe such regions [postgis.refractions.net]). Moreover, it can also be represented as a logical place. Logical places like "campus" or "bank" are mapped in the respective geographic by using a fixed dictionary. Logical places like "here" are mapped via simple algorithms considering the user current GPS location. Note that this enforces context-awareness, the same "here" information get multiple meanings depending on the user actual location.
- ***When*** is the time duration to which the context relates. It may be an exact range (e.g., "2006/07/19:09.00am - 2006/07/19:10.00am"), a concise description of a range (e.g., 9:28am), or even a logical value (e.g., "now", "today", "yesterday", "before"). Exact values are represented with a "begin-time-of-day – end-time-of-day" expression. Concise description and logical values are mapped via simple algorithms to the corresponding exact value. For example 9:28am = 2006/07/19:9:28am ± 5min. It is important to emphasize that concise time descriptions and logical times are contextual operators, their meaning depends on the time the query is actually issued.

Software agents are in charge of creating and inserting knowledge atoms in the shared space. Agents sense information from several devices (e.g. RFID tag, GPS devices, Web services) and combine them in order to produce a concise and effective description of what is happening in terms of a W4 tuple. The following examples illustrates the atom generation process.

Gabriella is walking in the campus' park. An agent running on her PDA can periodically create an atom describing her situation.

Who: user:Gabriella
What: works:pervasive computing group
Where: lonY, latX
When: now

The *Who* and *What* information are entered directly by the user at the login of the agent application, *Where* and *When* are dynamically provided by the GPS device.

Gabriella's PDA is connected with a RFID tag reader. A specific RFID agent controls the reader and handles the associated events. When a tag is read, the RFID agent creates a knowledge atom to store the tag information. In particular, either the tag would contain its own description, or the tag ID would be resolved in a dictionary to retrieve the description. This information, together with the "tag" namespace will fill the *Who* field. *What* is left unspecified. The agent accesses the GPS to retrieve location of the tag and fill the *Where* field. Finally, it completes the When field with the logical value "now".

Who: tag:statue of Ludovico Ariosto
What: -
Where: lonY, latX
When: now

C. W4 Interface

Knowledge atoms will be stored in a shared data space. In particular, our model relies on the following non-blocking and deterministic operations:

void inject(KnowledgeAtom a); enters a knowledge atom in the shared space
KnowledgeAtom[] read(KnowledgeAtom a); retrieves all the atoms matching a template knowledge atom.

The *inject* operation is trivial: an agent accesses the shared data space and store a knowledge atom there.

The *read* operation, instead, requires some more discussion. The W4 Model is suitable not only to represent context information, but for questioning too. A query will be represented by a W4 tuple with missing values (i.e., fields left unspecified). The read operation triggers a patter matching

procedure between the query and the knowledge atoms that already populate the data space. Matching atoms are returned as results of the query. In this process, it is important to understand that the pattern matching operations work rather differently from the traditional tuple space model. In fact, our proposal can rely on the W4 structure to enforce more expressive pattern matching operations that have a different meaning for the various Ws.

- **Who and What.** Pattern matching operations in these two fields is based on string-based regular expressions. For example, a patter like "user:*" will match any user.
- **Where.** Pattern matching in this field involves spatial operations (again inspired by Postgis operations). Basically, the template defines a bounding box. Everything within the bounding box, matches the template. For example, a pattern like "circle,center(lonY,latX),radius:500m" defines a circle centered at (lonY, latX) with a 500m radius. Tuples with a *Where* field within the circle will match the template. Logical places have to be translated into actual spatial regions before of going through the pattern matching.
- **When.** In this kind of pattern matching, the template defines a time interval. Everything that happened within that interval matches the template. Concise time descriptions and logical times will be converted into actual time interval before of pattern matching.

The following two examples illustrate the querying process.
Gabriella is walking in the campus, and wants to know if some colleague is near. She will ask (read operation):

Who: user:*
What: works:pervasive computing group
Where: circle,center(lonY,latX),radius:500m
When: now

Analogously, Gabriella can ask if some of her colleagues has gone to work in the morning:

Who: user:*
What: works:pervasive computing group
Where: office
When: 2006/07/19:09.00am - 2006/07/19:10.00am

It is important to emphasize that returned answers have not to be "complete" W4 atoms. The pattern matching mechanism also allows matches between incomplete information. Thus, following this approach, applications are based on components entering complete and incomplete context information and getting in response other refined (but possibly still incomplete) information.

D. Discussion and Future Extensions

In our opinion the proposed W4 model addresses some of the previous challenges. First of all, the model is general enough to be used in different application fields, e.g. outdoor and indoor navigation, location-based services etc. The field structure is suitable fore a variety of application, indeed no field is application specific. The 4-field structure is supposed to have meaning for almost all context subject. Nevertheless, if a field is empty the others may carry useful piece of knowledge. The lack of a field doesn't compromise the correctness of an atom neither the ability of retrieving atoms. This makes the model autonomic and adaptable. Even if the structure is very simple, it conveys a lot of information beyond the spatial ones. The description includes both parameters of the context (time, location) and parameters about activities being undertaken.
In our opinion, there are however two important extensions that could be valuably added to the model.
On the one hand, the current model is somewhat limited by the lack of a reference ontology that could add semantic relationships to the concepts in the W-fields. With such an ontology in place, knowledge atoms could be related also if their fields do not match exactly, and also application agents would be able to manipulate the retrieved context information in a more meaningful way.
On the other hand, Although pattern matching operations proved rather flexible to retrieve context information, in our future work, we would like to exploit the W4 structure to better navigate the context repository. More specifically, we would like to link together the various knowledge atoms to form a knowledge network where it would be possible to navigate from one W4 tuple to the other. From this perspective, the W fields could be link to other knowledge atoms, so that it would be possible, for example, to follow the *Where* link to get further information on where a given entity is located. Our idea, is that the possibility of querying this network, instead of a flat tuple space, would allow much more semantically rich questions and inferences. In particular, new knowledge could be produced by navigating the knowledge network and combining and aggregating existing information into new knowledge atoms.

IV. THE "BROWSING THE WORLD" INFRASTRUCTURE

To enable the concept of "browsing the world", we designed and implemented an infrastructure based on the W4 model. In this section we first present the general architecture underlying our infrastructure, then we will detail the parts that fulfill the W4 model.

A. The W4 Architecture

A general infrastructure to enable human-centric browsing of the world must include services for data acquisition, data integration, and data visualization. The architecture we have implemented is organized as follows:

1. Putting humans at the center, our architecture considers users with portable computing devices (i.e., laptops or PDAs), integrating localization devices (i.e., GPS), devices to acquire information from the physical world (i.e., RFID readers and sensors), and means to connect to the Internet (i.e., WiFi and/or UMTS connections).
2. Data coming from these devices (there included user GPS data) is represented by means of the W4 tuples, and stored in the local tuple space to be later accessed by application agents.
3. Relevant data are sent to a globally accessible shared tuple space containing the W4 model of the world. This space allows multiple users to exchange information and to conduct wide-area queries.
4. A RFID reader (in the form of a wearable glove) connected to the laptop or to the PDA via a serial cable

can be used to collect information from RFID tags dispersed in the environment. This information, enriched with the physical location where it has been collected (as provided by the GPS, device) is stored in the local tuple space.

5. Data coming from sensor network nodes (Crossbow MICAz) can be accessed by a suitable agent that collects sensed data and store them in the data space. Data is enriched with the physical location of the actual sensors and converted in the W4 format. Alternatively, sensor data could be collected by a base-station and sent directly to the "World" tuple space.
6. Specific services can be realized by means of application agents (i.e., autonomous software components) running locally on the user portable device and accessing, via the W4 model, both the local and "World" tuple spaces. Also, application agents can interface with a local GIS client (Google Earth or Google Maps) to turn data into a user-centric perspective.
7. Agents can dynamically connect to the Web to retrieve additional information to integrate with that coming from the W4 tuple spaces.

The whole system has been realized using the Java language. The "World" tuple space has been implemented through a Postgres database with spatial and temporal extensions. The local tuple space is simply implemented by a Java Vector. The RFID reader and the sensors are accessed via JNI and sockets respectively. User interface is provided by Google Earth (for laptops) and Google Maps accessed via the Minimo browser (for PDAs).

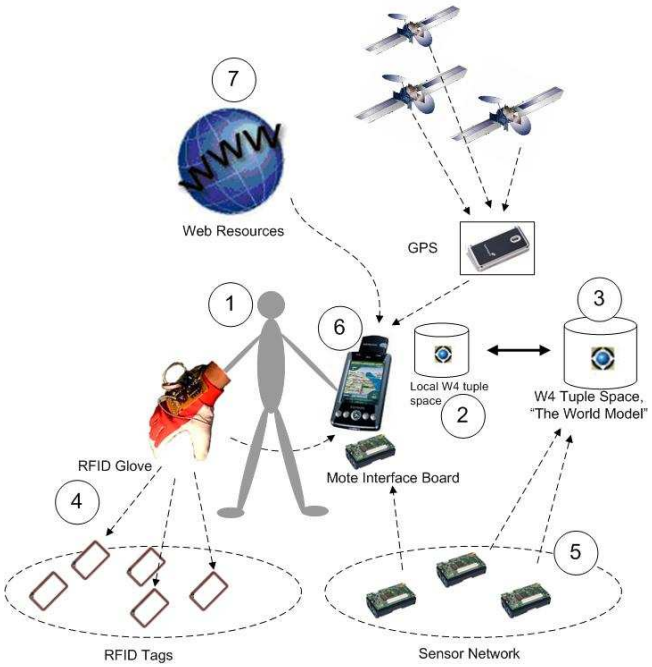

Figure 2. User centric infrastructure for browsing the world

B. W4 Tuple Space

All the information coming from the supported embedded devices (GPS, RFID and wireless sensors) is represented by means of W4 tuples, and stored in a local tuple space. Application agents access this space to retrieve W4 context information supporting their activities. Thus, application agents are completely decoupled from low-level embedded devices, and so they access and deal with contextual information only in tem of the W4 model. In addition, the availability of a local tuple space allows the system to work also in absence of a network connection and allows to minimize the generated data traffic (and its associated costs). Since this tuple space has to run on portable devices, it has been implemented by a simple Java Vector accessible with the W4 interface as described in 3.c.

Other than the local tuple space, our infrastructure is provided with a globally accessible shared tuple space containing a model of the world. This space allows multiple users to exchange information and to conduct wide-area queries. For scalability reasons, there could be more World tuple spaces, physically dislocated in the environment and linked in a DNS-like hierarchy. However, our current implementation consists of a single Web-accessible Tomcat server giving access to a Postgres database that store the W4 tuples. We realized JSP and Servlets implementing the W4 interface.

Application agents have to decide which information has to be sent to the World tuple space and which has to remain only locally confined. This decision may depend on many factor, such as privacy issues (e.g., a user may not be comfortable of constantly sending his GPS location on the Web) and scalability reasons. For example, trivial math says that storing one person entire life (100 years) GPS traces (2 floats) sampled at 0.1Hz amounts at 2.5 GB of highly redundant (thus compressible) data. Depending on needs, agents can decide the rate of data to send to the global server.

C. W4 Query Engine

The W4 query engine is the component that is in charge of managing the W4 queries and perform pattern matching operations.

The query engine running on the local tuple space has been developed in Java. It basically, scans the local Vector of tuples and uses String parsing methods and simple geometric algorithms (to handle *Where* clauses) for pattern matching.

The query engine running on the "World" tuple space dynamically translates W4 queries in SQL to execute them on the Postgres database. In this implementation, query pattern matching is supported either natively by SQL or by the Postgis spatial extension for the *Where* field.

It is worth emphasizing that the current implementation is only a first prototype and the current tuple space and query method is rather naïve. However, in future implementations, we will enrich the current infrastructure so as to manage, organize and integrate data in a more complex and clever way. In particular, as discussed in 3.4, we would like to abandon the current flat tuple-based implementation and structure context information in networks of knowledge. Such network-based representation would be more naturally distributable and could make our infrastructure more adaptive and autonomic.

D. The Graphical Interface

We developed a flexible graphical subsystem that can be easily employed on both laptops and PDAs. In particular, it interfaces with the GIS tools made available by Google: Google Earth and Google Maps to display retrieved context information as placemarks in a specific geographical area (see Fig. 3, 4, 5). Our graphical subsystem is based on the Keyhole Markup Language (KML), fully supported by Google Earth (at the moment only available for desktops and laptops), and at least partially supported by Google Maps and Google Maps for Mobile (that can be accessed also by PDAs and smart phones). This language allows to enrich geographical images coming from the Google GIS software with custom placemarks, images, 3D objects, etc. Thus, our graphical interface just translates proper W4 tuples in a corresponding KML file and dynamically provides it to the Google software. It is worth noticing that the KML language allows also to specify the user viewpoint on the map. This naturally supports context awareness, in that an agent could decide to center the map where relevant information are located.

Following this approach, application agents can then acquire relevant information by both interfacing with embedded devices, and relying on user interfaces. Such information will be represented in the W4 language for the sake of easy retrieval, manipulation and understanding. Finally, it will be converted in KML for the sake of effective visualization.

V. Application Examples

To test our model and infrastructure, we developed some simple applications highlighting the flexibility of the W4 model and infrastructure. In all these examples, we implemented a software agent that:

1. receives either static or dynamic queries from the user.
2. accesses the World tuple space to retrieve suitable context information.
3. creates a KML-formatted answer, and displays it either in Google Earth (for laptops) or in Google Map (for PDAs).

A. The Journey Map

A first application allows to provide context-aware information to a user equipped with a GPS device and a RFID reader. In particular, we focused on the scenario in which a tourist wants to automatically build and maintain a diary of his journey. To this end, the proposed service allows to keep track of all the user movements and have them displayed on the map of the visited place. Moreover, the support for RFID allows to access likely-to-be-soon-available tourist information stored in RFID tags attached to monuments and art-pieces. From the diary perspective, this allows to store the visited art-pieces' location together with their description on the journey map. The W4 model can accommodate a number of interesting queries in this scenario. A first query allows to retrieve information about RFID tags being read.

Who: rfid:*
What: *
Where: lonY, latX
When: now

We implemented this as a static query that the agent asks cyclically to the local cache of tuple space (recall that the RFID agent is the one in charge of reading nearby tags and represent them in W4 format). If a tag is found, its content (properly parsed and enriched with Web-retrieved information) is used to create a KML placemark that will be displayed in the user interface (see Fig. 3). It is worth noticing that data coming from sensor network could be accessed via a similar W4 query.

Another service we realized for the journey map application allows an agent to recover user past locations from the World tuple space. This service could be useful to review a past tour and check the places where the user has been. The associated W4 query can be expressed in the form:

Who: user:Gabriella
What: *
Where: *
When: yesterday

Similarly as before, we implemented this service as a static query. The agent queries the World tuple space, retrieves a list of past GPS traces and displays as a KML-ploryline in the user interface (see Fig. 5).

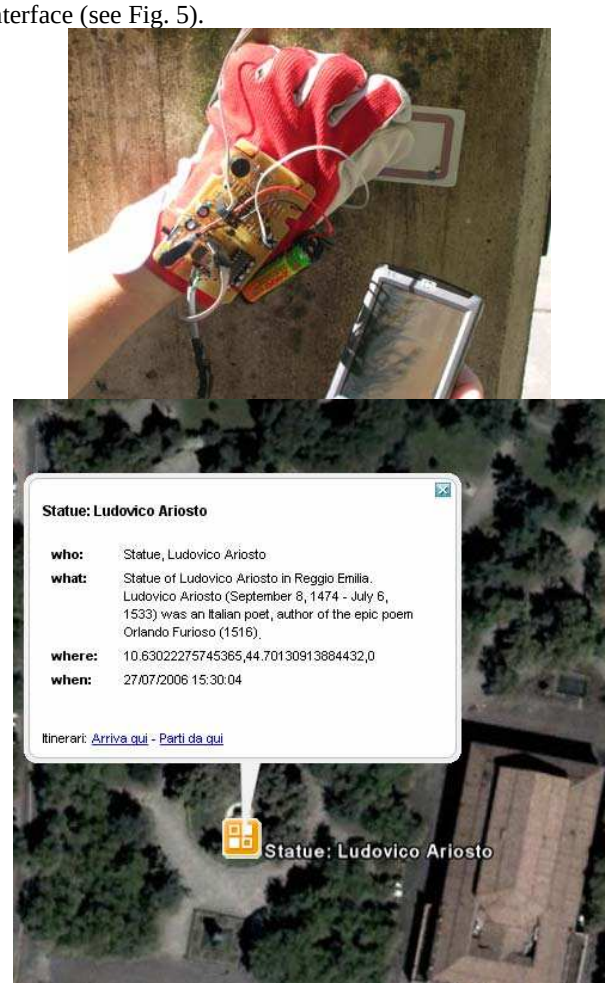

Figure 3. (top) The RFID-reader embedded in a glove allows to identify tagged objects. (bottom) RFID tags becomes placemark with Web-retrieved information in the GIS software.

Figure 4. GUI showing user's past GPS traces

B. The People Map

A user equipped with a GPS device can decide to share his location with other users and, analogously, he may wish to be aware of the location of others users. For example, a group of friends can share their actual GPS locations (represented as knowledge atoms) with each other. This can happen either by uploading knowledge atoms to the World repository, or by exchanging them in ad-hoc way and storing them in the local tuple space cache only. Either way, collected knowledge atoms can be used to display users' locations on real-time a map (which, by the way, can highlight other interesting Web-retrieved information for the group, such as museums or bar, depending on the specific interests of the group). It is finally worth noticing that our current implementation of the service deal with privacy by leaving up to the individual user to decide whether to: share its position or not (and with which accuracy), make it available only to a restricted group of users, or to make it publicly available but only in an anonymous way.

The W4 model can accommodate the some relevant tasks with the following query

Who: user:*
What: works:pervasive computing group
Where: *
When: now

In addition, using the W4 model, we developed an advanced interface to enable location dependent queries. A user can use the "Who", "What", "Where" and "When" fields to dynamically compose queries and to ask information about local facts and "things" (e.g., "Find all restaurants within 500 meters") and get in answer the visualization at the correct location (i.e., in the form of Google Earth / Google Maps placemarks) of all that is found matching the query. Since the answer to a location-dependent query is based on the location of the mobile users, the results of these queries dynamically change as the users change their location in context-aware fashion. The query interface lets the user choose the number of unknown fields, potentially the user can access to the whole atom knowledge letting all the fields set to "any".

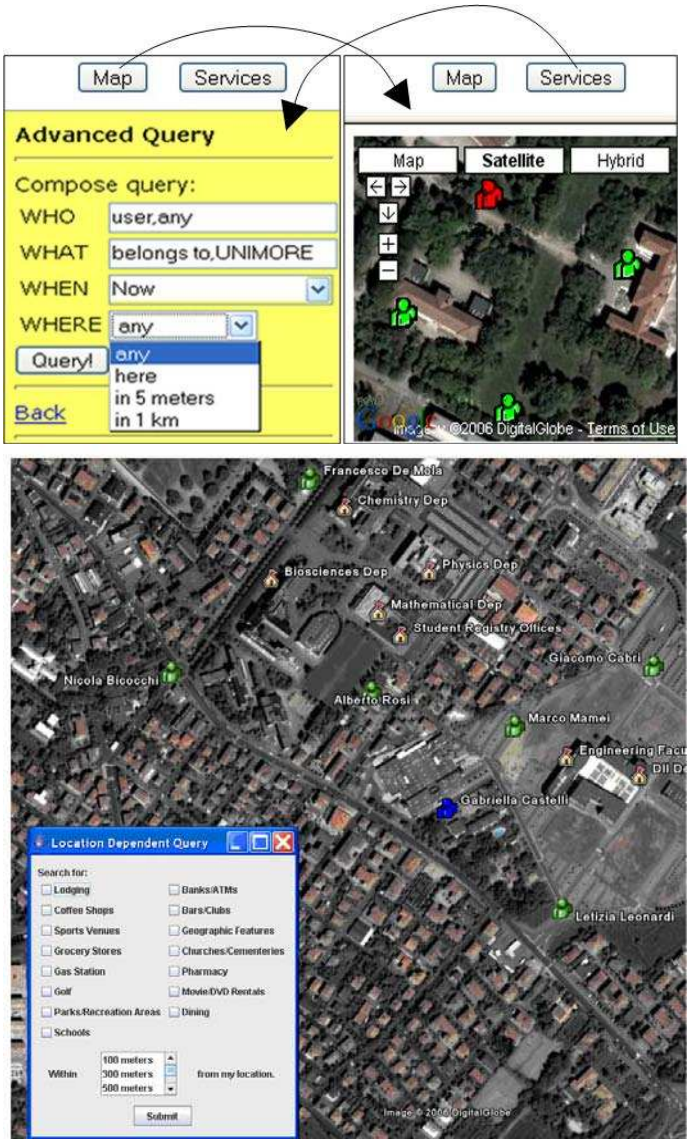

Figure 5. Map showing users real time locations with neighbor university facilities. (top) PDA user interface with the browser Minimo. (bottom) laptop user interface with Google Earth.

VI. Related Works

In the past few years, several models addressing contextual information and context-aware services have been investigated, and several infrastructure approaching -- to some extent – our concept of "browsing the world" have been proposed. In this section, we discuss and compare with ours some relevant proposals in these area.

A. Related Context Models

Existing researches on models for context-aware information try to create high-level and general-purpose context representation to be easily queried.

First works by Schmidt et al.[SchATT99] concentrate on the acquisition of context data from sensors and the processing of this raw data through a layered model. Similarly, the Context Toolkit [DeyAS99] focuses upon deriving context from raw

data by providing abstract component that can be connected together to capture and process the data from sensors. Although powerful, in our opinion, these approaches lacks of a common semantic to describe the data. This force developers to build new query languages depending on the kind of information at hand. On the contrary the W4 model provides a common semantic to deal with multiple context information in a coherent way.

We focuses upon develop a context model that can be easily queried. The pioneering work in this area is by Schilist et al. [SchAW94], who proposed a simple context model in which information are maintained by a set of environments variables. Analogously, Henricksen et al. in [HenIR02] analyze context adding the temporal aspect, information imperfection, various representation and high interrelation. However this approach leads to a long list of all characteristics of context, lacking in simplicity.

Others authors use structured context models as tuple space. In mobile computing several systems such as MARS[CabLZ00] and LIME[MurPR01] use the notion of reactive programming for shared unstructured tuple spaces. The Context Fabric' s fundamental abstraction is the InfoSpace [Hong02]. Each InfoSpace is a context tuple describing a single piece of context data in terms of entities (people, place, thing), attributes (e.g. the name) and relationship, special kinds of attributes that points to other entities. That approach doesn't address the temporal dimension of context. Egospaces [JulR02] provides a structured notion of context as name-value pairs in a Linda-like tuple space. Egospaces addresses context-aware programming in Ad-Hoc environments populated of agents by proposing an egocentric notion of context, i.e. every agent holds a personal representation of the world - that representation is called view. That approach is related to our approach with whom we share the idea of a structured representation of the context, however it is not concise because requires multiple tuples to represent a context piece. Indeed we don't have an egocentric notion of context.

Chang Xu et al. in [XuC05] propose a model very similar to our approach. It consists in a seven-field data structure that manage the description of the context. The fields are: subject, predicate, object, time, area, certainty, freshness, with similar meaning to W4. Beyond the field meaning, the purpose is different: their context model is not for browsing the world application. Similar considerations apply for the system described in [BraHCN06]: it describes RFID tags with the who-what-where-when structure. This approach is related to our with whom we share the idea of merging some information from different sources, e.g. the id from the tag with location from GPS or subject from the Web. However it's not a general model, since it is applied only to RFID tag.

B. Related Infrastructures

It is clear that a general infrastructures for browsing the world does not exist. Nevertheless, there exist several application-specific infrastructure and services which can show the importance of the problem. Some streams of works is simply based on representing Web information overlaid to geographical maps [But06, Rou05]. Examples include representations of avian–flu-outbreak reports [declanbutler.info/Flumaps1/avianflu.html], celebrity sightings [www.gawker.com/stalker], real estate information [www.forsalebyownercenter.com/google-earth-real-estate.aspx], and videogames [www.findskullisland.com]. Location-based services are natural candidates to use the power of novel GIS systems. A survey of novel tools to create location-based services is presented in [HarK05]. The presented systems combine GPS data, Web-search engines and GIS tools to retrieve and visualize services on a location basis. One system, for example, converts GPS data into street address by exploiting a standard geo-coding service (www.mapquest.com/features/main.adp?page=geocode). The address is then used to refine a search query submitted to a Web-search engine. Finally, the results are automatically displayed on a GIS tool. More dynamic applications, combine collaborative technologies (e.g., blogs and wiki) to GIS tools. MapWiki [TerK06], for example, is a Wiki collaborative environment where all the contents are located on a map. The contents can be edited and moved across the map and accessed on a location basis. Similarly, the Socialight software (socialight.com) allows a user to leave virtual Post-it notes, called sticky shadows, in specific sites around a city. The application checks the user actual coordinates with the note geospatial database and retrieves matching content.

In our opinion, all the above projects represents promising starting points of a future in which a wide range of information will be properly conveyed by novel GIS services. However, most of the above researches are special purpose and lack of a general architecture to manage and integrate pervasive, Web and GIS data. Furthermore, in opposition at our user-centric vision, their aim is to produce a centralized view of the world.

Other works concerned systems characterized by the presence of exploratory users and a surrounding environment. Users move forward the environment and access information exploiting different type of embedded sensors. Example of this systems are TinyLime [CuGG05] and all these system of world browsing like the system proposed in the past by our group [MamQZ06]. TinyLime is a middleware for wireless sensor networks that departs from the traditional setting where sensor data is collected by a central monitoring station, and enables instead multiple mobile monitoring stations to access the sensors in their proximity and share the collected data through wireless links. This context-aware setting is demanded by applications where the sensors are sparse and possibly isolated, and where on-site, location-dependent data collection is required. An extension of the LIME middleware for mobile ad hoc networks, TinyLime makes sensor data available through a tuple space interface, providing the illusion of shared memory between applications and sensors. At the same way our group describes the design and implementation of a tuple-based distributed memory realized with the use of RFID technology. The key idea is that everyday environments will be soon pervaded by RFID-tagged objects. By accessing in a wireless way the re-writable memory of such RFID tags according to a tuple-based access model, it is possible to enforce mobile and pervasive coordination and improve our interactions with the physical

world. From a certain point of view we can consider these systems as “World Browsing” systems. Nevertheless we can say that our new system is certainly more complete and structured. These systems indeed base their functionality on specific technologies (in this case environmental sensors and RFID tags), they aren’t based on a well structured standard context model and further they don’t exploit information coming from the Web (as our does). A further interesting project is FLAME2008 [WeissVoG04]. Users through their PDA access to services (related to the 2008 Olympics games in Bijing) expressly fitted on their needs: FLAME2008 elaborates them on the base of activities and situations carried out by the user. The infrastructure is very interesting, in particular for its use of ontologies, and appears to be very complete. Nevertheless we noticed that it’s too bound to a specific application field and it doesn’t perform any mechanism for generate and store new knowledge (think at our mechanism of generating new knowledge atoms from performed queries in the past). Our model is certainly more user centric and location independent, besides it has been developed to adapt itself to a generic context, and above all, to be fully functional in any location with or without infrastructure support.

VII. Conclusions and Future Works

In the next few years, browsing the world will be as common as today is browsing the Web, and the increasing number of proposals and applications in this area definitely testify this trend. However, a number of challenging research issues still have to be faced to fully realize the vision. In this paper we present a simple model and infrastructure to handle some of these challenges. Our future research in this area will mainly focus on two aspects. On the one hand, we will try to integrate ontologies in our model to improve its expressiveness and flexibility. In particular, ontologies will allow more semantic forms of pattern matching. On the other hand, we will try to go further than the current flat knowledge atom representation and link knowledge atoms in suitable knowledge networks allowing a better and more semantic navigation of context information.

References

[BraHCN06] J. Bravo, R. Hervas, G. Chavira ,S. Nava, "Modeling Contexts by RFID-Sensor Fusion," percomw, pp. 30-34, Fourth Annual IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOMW'06), 2006.

[But06] D. Butler, “Virtual Globe: the Web-Wide World”, Nature, 439:776-778, Feb 2006.

[CabLZ00] Giacomo Cabri , Letizia Leonardi , Franco Zambonelli, MARS: A Programmable Coordination Architecture for Mobile Agents, IEEE Internet Computing, v.4 n.4, p.26-35, July 2000.

[Cas06] G. Castelli, A. Rosi, M. Mamei, F. Zambonelli, “Browsing the World: Bridging Pervasive Computing and the Web”, 2nd International Workshop on Ubiquitous Information Systems, Munster (D), September 2006.

[ChoK03] C.-Y. Chong, S. P. Kumar, “Sensor Networks: Evolution, opportunities, and challenges”, Proceedings of the IEEE, 91(8):1247-1256, Aug. 2003.

[CuGG05] C.Curino, M. Giani, M. Giorgetta, A. Giusti, A.L. Murphy, G.Picco: “Mobile Data Collection in Sensor Networks: The TinyLIME Middleware”; Dip. di Elettronica e Informazione, Politecnico di Milano, Italy and Dept. of Informatics, University of Lugano, Switzerland.

[CugP] G. Cugola, G. P. Picco, "PeerWare: Core Middleware Support for Peer-To-Peer and Mobile Systems", Technical report available at the URL: http://peerware.sourceforge.net/

[DeyAS99] Anind K. Dey, Gregory D. Abowd, Daniel Salber. A Context-Based Infrastructure for Smart Environments. Proceedings of the 1st International Workshop on Managing Interactions in Smart Environments, Dublin, Ireland, Dec 1999.

[Esp01] F. Espinoza, P. Persson, A. Sandin, H. Nystrom, E. Cacciatore, M. Bylund, "GeoNotes: Social and Navigational Aspects of Location-Based Information Systems", International Conference on Ubiquitous Computing, Atlanta (GE), 2001.

[HarK05] R. Hariharan, J. Krumm, E. Horvitz, "Web-Enhanced GPS", International Workshop on Location and Context Awareness, Munich (DE), 2005.

[HeyIR02] K. Henricksen , J. Indulska , A. Rakotonirainy, Modeling Context Information in Pervasive Computing Systems, Proceedings of the First International Conference on Pervasive Computing, p.167-180, August 26-28, 2002.

[Hong02] Hong. J. I., “The Context Fabric: An Infrastructure for Context-Aware Computing.”, Proc CHI 2002

[JulR02] C. Julien , G. Roman, Egocentric context-aware programming in ad hoc mobile environments, Proceedings of the 10th ACM SIGSOFT symposium on Foundations of software engineering, November 18-22, 2002, Charleston, South Carolina, USA.

[MamQZ06] M. Mamei, R. Quaglieri, F. Zambonelli, “Making Tuple Spaces Physical with RFID Tags”, ACM Symposium on Applied Computing, Dijon, FR, April 2006.

[MasCE01] C. Mascolo, L. Capra, W. Emmerich, “An XML based Middleware for Peer-to-Peer Computing”, 1st IEEE International Conference of Peer-to-Peer Computing, Linkoping (S), 2001.

[MurPR01] Murphy A. L., Picco G. P., and Roman G.-C., “Lime: A middleware for physical and logical mobility.” In Proc. of the 21st Int’l. Conf. on Distributed Computing Systems, pages 524–533, 2001.

[Rou05]W. Roush, “Killer Maps”, Technology Review, 11 September 2005

[Sat05] I. Satoh, “A Location Model for Pervasive Computing Environments”, 3rd International Conference on Pervasive Computing and Communications, IEEE CS Press, pp. 215-224, March 2005.

[SchATT99] A. Schmidt , K. A. Aidoo , A. Takaluoma , U. Tuomela , K. Van Laerhoven , W. Van de Velde, Advanced Interaction in Context, Proceedings of the 1st international symposium on Handheld and Ubiquitous Computing, p.89-101, September 27-29, 1999, Karlsruhe, Germany.

[SchAW94] B. Schilit, N. Adams, and R. Want. Context-aware computing applications. In IEEE Workshop on Mobile Computing Systems and Applications, 1994.

[TerK06] Y. Teranishi, J. Kamahara, S. Shimojo, “MapWiki: A Ubiquitous Collaboration Environment on Shared Maps”, 6th International Symposium on Applications and the Internet Workshops, Phoenix (AZ), 2006.

[Wan06] R. Want, “An Introduction to RFID Technology”, IEEE Pervasive Computing, 5(1):25-33, 2006.

[WeissVoG04] N. Weißenberg, A. Voisard, Rüdiger Gartmann, “Using Ontologies in Personalized Mobile Applications”, GIS'04, November 12-13, 2004, Washington, DC, USA.

[XuC05] Chang Xu , S. C. Cheung, Inconsistency detection and resolution for context-aware middleware support, ACM SIGSOFT Software Engineering Notes, v.30 n.5, September 2005..