MDM 2006 JAPAN

7th International Conference on Mobile Data Management

May 10-12, 2006
Nara, Japan

IEEE COMPUTER SOCIETY
IEEE
LOS ALAMITOS, CALIFORNIA
WASHINGTON / BRUSSELS / TOKYO

Context integration for mobile data tailoring*

C. Bolchini, C. Curino, F. A. Schreiber, L. Tanca
Dipartimento di Elettronica e Informazione – Politecnico di Milano
Piazza Leonardo da Vinci, 32—20133 Milano (Italy)
{**bolchini,curino,schreiber,tanca@elet.polimi.it**}

Abstract

Independent, heterogeneous, distributed, sometimes transient and mobile data sources produce an enormous amount of information that should be semantically integrated and filtered, or, as we say, tailored*, based on the user's interests and context. Since both the user and the data sources can be mobile, and the communication might be unreliable, caching the information on the user device may become really useful. Therefore new challenges have to be faced such as: data filtering in a context-aware fashion, integration of not-known-in-advance data sources, automatic extraction of the semantics.*

We propose a novel system named Context-ADDICT *(*Context-A*ware* D*ata* D*esign,* I*ntegration,* C*ustomization and* T*ailoring) able to deal with the described scenario. The system we are designing aims at tailoring the available information to the needs of the current user in the current context, in order to offer a more manageable amount of information; such information is to be cached on the user's device according to policies defined at design-time, to cope with data source transiency.*

This paper focuses on the information representation and tailoring problem and on the definition of the global architecture of the system.

1. Introduction

Today we are living an epochal change, whereby the advent of the internet and the development of the communication technologies have completely changed the focus of the information retrieval, from the struggle for finding information and organizing it to that of filtering the enormous stream of available data.

Mobility is, at the same time, becoming crucial for people, emphasising old challenges while bringing to the surface new ones. As a consequence, the context the user is acting in becomes more and more relevant for information filtering, and new context-aware applications begin to appear.

Furthermore, information sources themselves are changing: with the advent of peer-to-peer networks, file sharing, blogging, podcasting, and technologies such as sensor networks and web services, the concept of data source is mutating, assuming a far larger meaning with respect to the centralized static data sources such as legacy databases. The development of the Semantic Web [2] has unveiled the need to move the level of web data management from a purely syntactic level to a more semantic one [10].

We are thus dealing with a scenario which sees independent, heterogeneous, distributed, sometimes transient and mobile data sources produce an enormous amount of information that should be semantically integrated and filtered, or, as we say, *tailored*, based on the user's interests and context. Since both the user and the data sources can be mobile, and the communication might be unreliable, caching the information on the user device may become really useful. While the traditional problems typical of the data integration field are far from being solved, new challenges have also to be faced such as: data filtering in a context-aware fashion, integration of not-known-in-advance data sources, automatic semantics extraction.

A lot of these issues are common to several applications, so they should be eploited by an appropriate middleware to offer common services to the applications, and applications themselves have to be designed taking into account the above described scenario; thus, appropriate guidelines and methodologies need to be developed.

We propose a novel system named *Context-ADDICT* (*Context-A*ware *D*ata *D*esign, *I*ntegration, *C*ustomization and *T*ailoring) able to deal with the described scenario. Such an approach stems from our previous work on modeling the user's interests and context [5], but it leverages the result of the ontology field to catch the semantics implicitly or explicitly[1] laying in the data sources. A design method-

*This reasearch is partially supported by the FIRB project MAIS.

[1] In the following we will treat separately data sources exposing explicitly or implicitly the data semantics.

ology is currently under development as well: it is an evolution of the relational *Very Small DataBase Methodology* presented in [6], and we refer to this new methodology as *Ontology-based VSDB Methodology*. The Ontology-based VSDB Methodology aims at assisting the application designer to capture user's needs and the expected application contexts in a context and user interest model that we call *Ambient Dimension Tree*. This information, together with a *Domain Ontology* modelling the main aspects of the application domain is exploited to semantically integrate disparate data sources, offering the user a unique, semantically coherent, view over all available and interesting data. The system we are designing *tailors* the available information to the needs of the current user in the current context, to offer a more manageable amount of information to be cached. This goal is pursued, according to policies defined at design-time, by caching data on the user's device to cope with data source transiency. This operation fulfills the need for data filtering described above and matches the resource constraints that a mobile user device may have; see [4] for a discussion of resource constraints on small portable devices. An early tailoring approach, applied to the schema level, guarantees the system to operate on the minimum possible amount of data: the context-and-user-aware set of relevant data.

The exploitation of ontologies offers the chance to access data at a more semantic level, enabling, together with standard database queries, the execution of articulated reasoning tasks typical of description logics and knowledge bases. Although, at the moment, shared and agreed ontologies are not very common, apart from specific fields such as medicine, we believe that, with the advent of the *Semantic Web*, their diffusion will increase, and that communities on the web will converge on commonly agreed and formal ontologies. Therefore, we find them useful means to provide a common, well formalized, XML-based format for the description and integration of data source schemata.

Potential applications and scenarios for such a system range from centralized data management in a traditional intranet to mobile P2P information sharing: while a semiautomatic, precise semantics management can be the core of the first scenario, a dynamic mobile behavior of the data integration system is a key issue in the second one, thus the proposed architecture must be adaptable and tunable.

This paper describes the first step of this research: in particular, we chose to focus on the information representation and tailoring problem and on the definition of the global architecture of the system. These two issues, in fact, are among the most challenging parts of the work and clearly the first ones to be considered: indeed a general architecture can give us the chance to develop the various modules concurrently and to early evaluate the feasibility of the overall system, while the information representation and tailoring subproblem is of key importance in such a data centric system and influences almost all the other components.

2. Related Work

Much research is being carried out in the above mentioned field, both in the direction of complete systems and in the development of methodologies and tools dealing with one or more of the issues we are considering. However, we believe that, no solution to the full problem has been proposed yet. Among the most representative approaches we find: *InfoSleuth* [24], an agent-based query processor, *ONION* [23], an ontology integration framework, *OBSERVER* [22], an interesting data integration system, *MOMIS* [1], which approaches the integration of heterogeneous data sources using a global ontology, *INFOMIX* [16], an integration system based on computational logic, *SOCAM* [14], an interesting context model, *MAIS* [18], a project focused on Multi Channel Adaptive Information Systems. All of these systems provide interesting approaches to partially solve the problem we are considering. The main differences between our approach and the mentioned ones are the following: most of the systems focus on the integration of ontologies containing the actual data as instances, on the other hand, the ontologies are exploited in *Context-ADDICT* only to *integrate the schemas*, thus the integration is only the first step of our work; our original contribution concerns what we call *Data Tailoring*. Most of the existing solutions do not take into account mobility, device resource constraints, and few of them consider the issue of context-awareness, while our goal is to offer a fully automatic integration and context-aware tailoring of not-known-in-advance *Data Source*s. Among the most interesting techniques and tools for semantic extraction and ontology merging we considered: *Ontolift* [29], *ERONTO* [28], *Relational.OWL* [26], *GLUE* [11], *Chimaera* [20], *ER2WO* [30], *QOM - Quick Ontology Mapping* [13], *FCA-Merge* [27], *PROMPT* [25]. In our opinion no one of these approaches provides an easily extensible and integrable solution to the problems we are considering, since most of them require a lot of user interaction and are rather power hungry applications. In this sense we are trying to build a suite of tools integrated in the *Context-ADDICT* system providing a lightweight solution to issues such as semantic extraction and ontology mapping. Other interesting related works are the precise formalism of [8], which provides important theoretical foundation on information format translation, between ER and OWL [21] and *SOCAM* [14], a complete context model. A much more detailed analysis of the related work of the *Context-ADDICT* system is provided in [7].

3. Context driven information management

The aim of our project is the design of a framework supporting the development of context-aware and data centric applications from the design phase down to the system deployment, focusing on semantical integration and data filtering, with particular interest for distributed, potentially peer to peer, applications, where mobility and context awareness become key issues.

More precisely, the focus is on a data-centric notion of context, in the sense that its features are based on the data the mobile device must store in order to autonomously support a specific application in a given context. We also propose effective tailoring procedures to be used to dynamically integrate and filter the data offered by heterogeneous and independent data sources. We consider data filtering a necessary task both to fulfill the need of the user to obtain only the relevant information, and to match the user device constraints that in a mobile, storage and power-aware scenario may play a key role.

In order to attain our goal, the following functionalities are needed:

- a complete *user and context model*;
- Data Source discovery service;
- semantical (semi)automatic integration of Data Source schemata;
- *Data Tailoring*: semantical data filtering based on the user and context model;
- support for distributed query processing;
- support for data synchronization.

Through the proposed system, the user gains access to a context-aware global view of the available and interesting data, which can be semantically queried independently of the actual data format (Relational, XML, web pages, web services, data from sensor networks...).

3.1. Context Model

The Ambient Dimension Tree is an advanced user profile and context descriptor based on the concept of *dimension*, an extension of the context and user interest model presented in [6]. It is used to represent in a ontology-based format the description of the user needs, and to capture the context the user is acting in. In [5] and [6] it has been shown how the designer can exploit the concept of dimension to capture the different characteristics of a user profile and of the context of an application, but while the model in [6] was array-based, in this work a tree-based ontological model has been developed, both to enrich the designer's chance to define contexts and to enable more automated ways of operating with this object.

The designer, guided by a dedicated methodology, which is currently under development, will define a set of dimensions used to describe the context and the user profile. A dimension captures an aspect of a context or of a user profile. In our experience, there are some dimensions that often occur in the applications: here we list the most common ones in our opinion, although it may happen that only a subset of them be needed, or that other dimensions come into play.

- Interest Topic: The various topics of interest for the possible users of an application.
- Holder: The various user categories involved.
- Situation: Different points of view for a same interest topic, for example "hospitalized" and "at-home" for a medical personal care application.
- Space: Based on the place where the user is currently located (it might be GPS coordinates or any other location information), its granularity may vary depending on the application. For example a museum's "room" or an whole "city" or a "region".
- Time: a temporal indication based on the current time, its granularity may vary. For example, one could choose the current "month" or the "last year", but the reference point is always *now*.
- Ownership: a dimension describing the access rights to the actual data, is used to enforce privacy and security issues that may come into play.

As said before, not all *dimension*s are always necessary and more might be needed. It will be the designer's task to establish, following the methodology guidelines, which dimensions are appropriate for the application he/she is designing.

For each dimension, the designer defines a set of admissible values called *Dimension value*s, defined as follows: a dimension value is an instantiation of a certain dimension concept: an indication capturing a precise value of an aspect of a context, e.g., for the dimension *holder*, a Dimension value can be *"student"* or *"employee"*, while the dimension *space* may have as values space granularities such as *"city"* or *"campus"*.

By means of dimensions and dimension values, the Ambient Dimension Tree describes all the possible users and contexts, i.e., the dimensions define a multidimensional space, where each point in the space represents a potential user profile and context.

To capture a single point in the context-user multidimensional space we use the concept of *Ambient Dimension Tree*

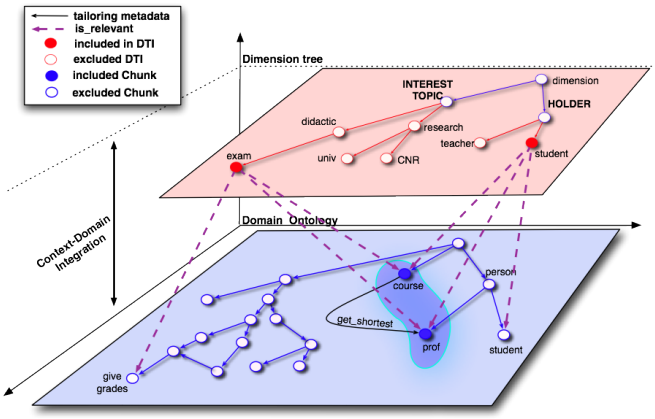

Figure 1. Design time *Chunk* definition.

Instantiation: a set of dimension values completely specifying all the dimensions of an Ambient Dimension Tree[2]. An Ambient Dimension Tree Instantiation (DTI) fully defines a precise user profile and context, we can say that precisely defines a point (or a subspace) in the context-user multidimensional space, for example *"professor" - "free room" - "exam session" - "campus" - "this month"* represents a professor interested in rooms which are free this month during the examination period (as a situation) in the campus he/she is located in.

We will show in Section 3.3 how each one of the multidimensional points will be related to a set of relevant concepts of the domain, thus a set of relevant data in the Data Sources. We will refer to this set of relevant data as a *Chunk*.

3.2. Domain Model

The domain model is the Domain Ontology, a general ontology formally describing the main concepts of the application domain and their relations. The use of an ontology as a domain model offers the chance to build a commonly agreed dictionary to understand the domain, useful to automatically integrate the data sources, and will enable the user to semantically query the data. This ontology may be already existing and should cover all the relevant concepts and relations of the application domain. This ontology is instanceless, since instances will be retained from the actual data sources after the integration and tailoring phases.

3.3. Context-Domain relationship

The Ambient Dimension Tree and the Domain Ontology become useful tools when properly used in combination. By wiring them together, it is possible to capture which portion of the domain is actually relevant for each user-context pair: the relevant chunk of data. The chunks in [6] were manually defined by the designer as relational views over a global schema, while here, in order to deal with not known in advance Data Sources and to make the process more automatic, we designed a new way to define the chunks based on the Enhanced Domain Ontology. The Enhanced Domain Ontology is used to capture the domain, the context and their relationships together, as defined by the designer; it thus contains the definition of the chunks associated with every possible Ambient Dimension Tree instantiation. The Enhanced Domain Ontology is an ontology containing:

- the Ambient Dimension Tree: to model the context and user profile;
- the Domain Ontology: to describe the domain;
- the *Context-Domain Integration*: an ontology which captures the relationship between the domain model and the context model.
- the *Tailoring Metadata*: an ontology used to capture a set of policies to deal with design-time unknown concepts (the one introduced by the Data Sources and not included in the Domain Ontology).

While the Ambient Dimension Tree and the Domain Ontology have already been discussed, we now focus on the Context-Domain integration and the Tailoring Metadata.

The Context-Domain integration relates the elements of the Ambient Dimension Tree to those of the Domain Ontology, as shown in Figure 1. This is done by exploiting a particular role named *is_relevant*. Each dimension value is related, via the *is_relevant* role, to the set of Domain Ontology concepts relevant for it. Therefore, given an Ambient Dimension Tree instantiation, it is possible to define which portion of the Domain Ontology is relevant: the chunk corresponding to the Ambient Dimension Tree instantiation. A concept is included in the chunk if it is relevant for, at least one dimension value from each dimension. Because of the integration, the set of relevant data is extended to the concepts coming from the Data Source; this is done by following general policies and via the Tailoring Metadata.

The Tailoring Metadata will be used to define tailoring policies. These policies are used to deal with design-time unknown concepts, coming from the Data Source. Consider as an example, shown in Figure 1, the *get_shortest* policy: all the concepts/relations that will appear after the integration on the shortest path between *"course"* and *"professor"* should be included in the chunk. General topological rules are used to express these tailoring policies, this is part of an ongoing analysis work.

[2] At least one *Dimension value* for each dimension should be taken.

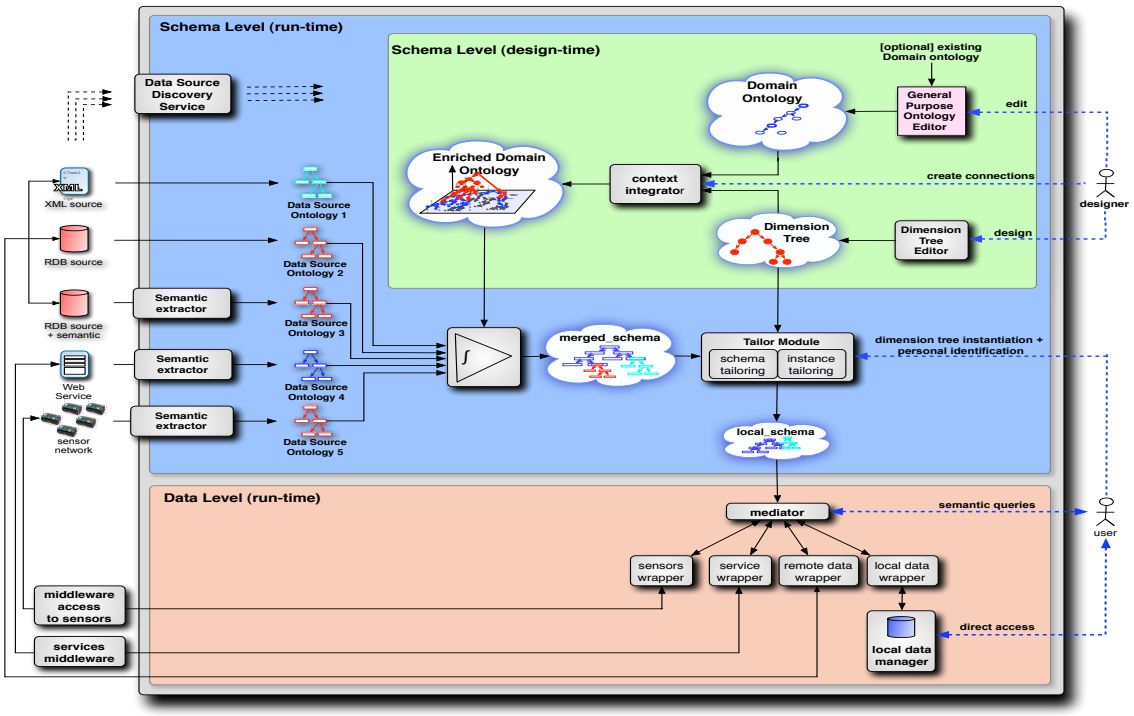

Figure 2. The proposed architecture.

4. The *Context-ADDICT* System

In this section we present the overall architecture of the system. We devote some sections to present both a sketch of the design methodology we are defining, and a deeper insight of each one of the components. The proposed architecture is shown in Figure 2. It is possible to logically partition the overall system into some subsystems, each one devoted to a specific task:

- The Schema Level subsystem further decomposed into:
  - Design-time subsystem
  - Run-time subsystem
- The Data Level subsystem.

The *Design-Time Schema Level subsystem* is typically used by the designer to define the Domain Ontology, to design the system behavior and to set up all the necessary configurations. The *Run-Time Schema Level subsystem*, composed of several modules, performs schemata integration and filtering; the *Data Level subsystem*, once the schema has been integrated and tailored, is devoted to the actual data movement, to synchronizing and integrating the actual instances of data only for the pieces of information considered relevant, and to user query processing (both local and remote). In the following, to complete the global vision of the system, a detailed description of these subsystems is presented, whereas the last subsection presents, the kinds of Data Sources we have foreseen.

4.1 Schema Level subsystem: design-time

At design-time, the designer operates by means of the *General Purpose Ontology Editor*, to modify an existing Domain Ontology or to design it from scratch (see the upper-right corner of Figure 2). The Domain Ontology captures the main concepts of the application domain; we expect it to provide a well accepted general taxonomy enriched by the main relations among concepts, even if not all the concepts attributes and relations are detailed. It should be a skeleton used as a guide to integrate the detailed Data Source Ontologies. We can say that this ontology answers the question: *"How is the world we are dealing with?"*. The Ambient Dimension Tree, on the contrary, captures a schematic representation of the user interest and context. The Ambient Dimension Tree is used to represent the knowledge of *"What may the user want to do in this domain?"*. The Ambient Dimension Tree is created by the designer via the *Dimension Tree Editor*, a simple syntax oriented ontology editor enforcing the Ambient Dimension Tree proper structure. The tool guides (by following a precise methodology) the designer in the process of capturing

context and user profile.

At design time the designer will use the Ambient Dimension Tree to describe the main features of the context and user interests, and later, via the Context Integrator module, he/she will capture the relationship between the Ambient Dimension Tree and the Domain Ontology. It is worth noting that, although both elements are represented via ontologies, this operation is not the standard mapping or merging of ontologies. The connections the designer draws via the Context Integrator module represent in general the relevance of Domain Ontology concepts for parts of the Ambient Dimension Tree, thus knowledge about *"What part of the domain is relevant for a given user and context?"*. This process is heavily application dependent, and cannot be performed automatically in any sense; this means that the Context Integrator must be a highly interactive tool, meant to give the designer an important role, as shown in Figure 2. Setting the relationship between the context and the Domain Ontology means to be able to capture, for each user and context, what portion of the Domain Ontology is relevant, so that the actual data will be properly tailored later on to provide users with the set of data relevant to them in a given context, discarding unnecessary information.

By capturing the correspondences between the Ambient Dimension Tree context model and the Domain Ontology we can tailor the Domain Ontology operating choices only at the Ambient Dimension Tree level. The result of this operation performed at design time is the Enhanced Domain Ontology, which captures both domain and application needs.

4.2 Schema Level: run-time operations and modules

On the leftmost side of the system scenario we find the Data Sources which might be fully heterogeneous in terms of schemata, data format and access interfaces. They may range from Relational Databases to XML data sources, to Web Services, to sensor networks, but the key point is that it is always possible to trasform their schemata into an ontological format and then operate on a uniform format, as shown in [8]. Some of these Data Sources will be *cooperative*, i.e., they will provide autonomously an ontological description of the available data, while others may offer a raw interface such as the DDL specification of the database schema. In the latter scenario we will automatically translate the schema into an ontological format (see details in [17]). This means that an automatic semantic extractor for each type of Data Source is necessary. Such a need is represented in Figure 2 by the *semantic extractor* boxes between the Data Sources and the corresponding Data Source Ontology. At run time the *Data Source Discovery Service* will be in charge to actually discover Data Sources and make them reachable. This module may heavily vary depending on the single scenario we are considering, from a centralized server to a set of fully distributed discovery procedures (see [9] for a detailed discussion), from a mere syntactic matcher to a semantical filter.

In general, data sources may appear and disappear during system working time; however, considering a snapshot of the system, at a given time a set of Data Source Ontologies and an Enhanced Domain Ontology are available; thus a standard ontology integration is needed. This operation is performed at run-time either on the user's device or on a dedicated machine, depending on the deployment choices of the designer. The integration operation is rather general and a well known problem, though far from being solved. A lot of research effort has been devoted to make this process as automatic and precise as possible (see [12] for a survey). In this sense we plan to leverage the literature results to provide an efficient *Integration Module* with good, *precision* and *recall*, performance, and able to work on potentially low power devices.

At the end of the integration the *Merged Schema*, as we have called it, is produced. It contains all the Data Sources information coherently integrated in the Domain Ontology view of the world. Since the correspondences drawn between the Ambient Dimension Tree and the Domain Ontology are still available, the result is a global schema instrumented with the context and user interest model of the Ambient Dimension Tree. At this point the integration task is completed and the data tailoring part comes into play: i.e., the innovative aspect of the proposed approach. To reduce the amount of data to be managed on the user device and to allow the user to access only the local portion of the system, the Ambient Dimension Tree context model is to be instantiated with actual values describing the current user interest and context. As mentioned before, by exploiting the correspondences drawn by the designer assisted by the Context Integrator module, the system can select the relevant concepts of the Domain Ontology; since the Data Source Ontologies have been integrated in the Domain Ontology, it is now possible to select also those concepts from each Data Source Ontology that are relevant for the given user in the given context, thus understanding, which portion of data from each Data Source is relevant to the user. This operation is performed over the Merged Schema by the Tailor Module. The second step of this operation foresees a more precise filter of the data, based on user identification. For instance, a student (user) may be interested in the exam rooms for the courses he/she attends and not in rooms for all the courses. This is performed by the rightmost portion of the Tailor Module exploiting a set of user identifier indications provided in the Ambient Dimension Tree plus the actual values inserted by the user during initialization. The result is a Local Schema, containing only the portion of the

Merged Schema relevant to the user. During the described process, several metadata have been recorded, together with the data schema, in order to enable query processing and synchronization as described in the next section.

4.3 Data Level subsystem

The *Data Level* subsystem deals with the actual data transfer. Though it has not been deeply investigated yet, we plan to apply here the results of preexisting researches on distributed and heterogeneous query processing (see [15, 19] for surveys). Each Data Source Ontology, where needed, will contain a set of metadata containing a description of how each concept of the the Data Source has been stored. For example, an annotation informing the system that the concept "teacher" of the Data Source 1 is stored in a table named "professor" of a database named "university" reachable at a given URL. This metadata has been tailored by following the same process; basically only the metadata about the concepts considered relevant have been included in the Local Schema. It is worth noting that the Local Schema may contain portions of data coming from several sources, so the system will split the query into two parts: reasoning and retrieval, the latter will be then divided into a set of one-data-source queries, translated from the ontological format to the local Data Source format. The metadata information, given a query over the Local Schema, supports its translation into a set of queries over the Data Sources in the appropriate languages.

The Data Sources will then process the queries; the returned result will be the base of the reasoning task. However, since in the most general case the data might be stored in transient Data Sources, and the network connection may be expensive or unreliable, part of the data will be stored on the user's device, to be always available. So, together with an on-line query processing, the Data Level Subsystem is responsible of the data synchronization and local data management. Because of the tailoring phase, the last two operations are performed on a manageable amount of data which are actually relevant to the user. A Local Data management service such as the one presented in [3], as well as mechanisms for data synchronization are required together with a set of caching policies. Apart from the caching and storage policies, the process of synchronizing the data will be similar to query processing, with the addition of local data memorization. A final remark related to the Data Level Subsystem is that the mediator-wrapper subsystem may interact with appropriate middlewares to access some of the Data Sources. The next section introduces, in more details, the kind of Data Sources we have to deal with.

4.4 Data Sources

In the considered scenario there are a lot of heterogeneous data sources that become available during the system lifetime. These sources may vary from Relational Databases, to XML files, from Sensor Networks, to Web Services or web pages, from peers of a P2P network to any other kind of information sources, as shown in Figure 2. As a consequence, the integration process is complex, mainly because the Data Sources are not known in advance and they may be transient due to user or Data Source mobility. Therefore, from the system point of view, it is possible to classify the Data Sources into the following categories: *cooperative* and *non-cooperative*;

- *Cooperative*: this kind of Data Sources are fully compatible with the system, because they offer an ontological view of the data, and they may even share the same Domain Ontology. Although such sources may internally represent data in any format, they expose an ontological description of their schemata. This means that the administrator of the Data Source has decided to design and publish an ontology that describes data available in that Data Source: the Data Source Ontology. This kind of Data Source will also provide the wrapper for its data so that the overall system will issue queries directly over the Data Source Ontology.

- *Non-cooperative*: since it is not possible to assume that all Data Sources will be cooperative as described above, especially at the present time, Data Sources that simply expose the schema in their native format need be taken into account. For example a relational Data Source will provide the DDL data dictionary; in such a case a run-time automatic ontology extraction should be considered. The subsequent integration and tailoring tasks will be performed as in the *cooperative* case.

Independently of the type of Data Source, there will be an ontology describing the data and all the metadata needed for query processing. Some of the cooperative Data Sources might be integrated at design time; in such a case we refer to them as *negotiated* sources.

5. Conclusions

The *Context-ADDICT* system faces a very challenging scenario where distributed, heterogeneous, independent, maybe mobile and transient data sources come into play. The goal is to automatically integrate and tailor the available data, based on a context and user model named Ambient Dimension Tree, in order to provide users with the set of relevant data.

The ultimate goal of *Context-ADDICT* is to support specific applications with an appropriate integration and tailoring layer, offering the application designer the chance to focus on the business logic. A design methodology, currently under development, will guide the application designer to properly exploit the *Context-ADDICT* system.

Together with the general definition of the system architecture we have focused the attention on the representation of all the information the system should manipulate, a key point in such a data-centric system. We have devoted particular attention to the formalization of the context and domain models.

References

[1] S. Bergamaschi, S. Castano, M. Vincini, and D. Beneventano. Semantic integration of heterogeneous information sources. *Data Knowl. Eng.*, 36(3):215–249, 2001.

[2] T. Berners-Lee, J. Hendler, and O. Lassila. The semantic Web. *Scientific American*, 284(5):34–43, May 2001.

[3] C. Bolchini, C. Curino, M. Giorgetta, A. Giusti, A. Miele, F. A. Schreiber, and L. Tanca. Polidbms: Design and prototype implementation of a dbms for portable devices. In *Proc. of the Twelfth Italian Symposium on Advanced Database Systems, SEBD, S. Margherita di Pula, Cagliari, Italy*, pages 166–177, 2004.

[4] C. Bolchini, F. Salice, F. A. Schreiber, and L. Tanca. Logical and physical design issues for smart card databases. *ACM Trans. Inf. Syst.*, 21(3):254–285, 2003.

[5] C. Bolchini, F. A. Schreiber, and L. Tanca. A context-aware methodology for very small data base design. *SIGMOD Record*, 33(1):71–76, 2004.

[6] C. Bolchini, F. A. Schreiber, and L. Tanca. A methodology for very small data base design. *Information Systems*, to appear.

[7] C. A. Curino. Context aware integration for mobile data design. Master's thesis, Politecnico di Milano - Dipartimento di Elettronica e Informazione, 2005.

[8] D. Calvanese, M. Lenzerini, and D. Nardi. Unifying class-based representation formalisms. *Journal of Artificial Intelligence Research*, 11:199–240, 1999.

[9] A. Celentano, F. A. Schreiber, and L. Tanca. Requirements for context-dependent mobile access to information services. In *Proc. 10th International Workshop on Multimedia Systems (MIS 04)*, pages 60–65, 2004.

[10] B. Chandrasekaran, J. R. Josephson, and V. R. Benjamins. What are ontologies, and why do we need them? *IEEE Intelligent Systems*, 14(1):20–26, 1999.

[11] A. Doan, J. Madhavan, P. Domingos, and A. Halevy. Ontology matching: A machine learning approach, 2003.

[12] M. Ehrig, J. de Bruijn, D. Manov, and F. Martn-Recuerda. D4.2.1 state-of-the-art survey on ontology merging and aligning v1. Technical report, Institut AIFB, Universität Karlsruhe, 2004.

[13] M. Ehrig and S. Staab. Qom - quick ontology mapping.

[14] T. Gu, H. K. Pung, and D. Q. Zhang. A service-oriented middleware for building context-aware services. *J. Netw. Comput. Appl.*, 28(1):1–18, 2005.

[15] D. Kossmann. The state of the art in distributed query processing. *ACM Comput. Surv.*, 32(4):422–469, 2000.

[16] N. Leone, T. Eiter, W. Faber, M. Fink, G. Gottlob, and G. Greco. Boosting information integration: The infomix system. In *Proc. of the Thirteenth Italian Symposium on Advanced Database Systems, SEBD, Bressanone, Bolzano, Italy*, pages 55–66, 2005.

[17] P. Lino and A. Rodiani. Trattamento in owl di sorgenti di dati relazionali: il caso del sistema per le informazioni dei musei basato su palmari. Master's thesis, Politecnico di Milano - Dipartimento di Elettronica e Informazione, 2005.

[18] MAIS. Mais: Multi channel adaptive information system project http://black.elet.polimi.it/mais/index.php.

[19] N. Marsit, A. Hameurlain, Z. Mammeri, and F. Morvan. Query processing in mobile environments: A survey and open problems. In *DFMA '05: Proceedings of the First International Conference on Distributed Frameworks for Multimedia Applications (DFMA'05)*, pages 150–157, Washington, DC, USA, 2005. IEEE Computer Society.

[20] D. L. McGuinness, R. Fikes, J. Rice, and S. Wilder. The chimaera ontology environment. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*, pages 1123–1124. AAAI Press / The MIT Press, 2000.

[21] D. L. McGuinness and F. van Harmelen. OWL Web Ontology Language Overview, W3C Recommendation, 2004.

[22] E. Mena, V. Kashyap, A. P. Sheth, and A. Illarramendi. OBSERVER: An approach for query processing in global information systems based on interoperation across pre-existing ontologies. In *Conference on Cooperative Information Systems*, pages 14–25, 1996.

[23] P. Mitra, G. Wiederhold, and M. Kersten. A graph-oriented model for articulation of ontology interdependencies. *Lecture Notes in Computer Science*, 1777:86+, 2000.

[24] M. H. Nodine, J. Fowler, and B. Perry. Active information gathering in infosleuth. In *CODAS*, pages 15–26, 1999.

[25] N. F. Noy and M. A. Musen. The prompt suite: interactive tools for ontology merging and mapping. *Int. J. Hum.-Comput. Stud.*, 59(6):983–1024, 2003.

[26] C. Pérez de Laborda and S. Conrad. Relational.OWL - A Data and Schema Representation Format Based on OWL. In S. Hartmann and M. Stumptner, editors, *Second Asia-Pacific Conference on Conceptual Modelling (APCCM2005)*, volume 43 of *CRPIT*, pages 89–96, Newcastle, Australia, 2005. ACS.

[27] G. Stumme and A. Maedche. FCA-MERGE: Bottom-up merging of ontologies. In *IJCAI*, pages 225–234, 2001.

[28] S. R. Upadhyaya and P. S. Kumar. Eronto: a tool for extracting ontologies from extended e/r diagrams. In *SAC '05: Proceedings of the 2005 ACM symposium on Applied computing*, pages 666–670, New York, NY, USA, 2005. ACM Press.

[29] R. Volz, D. Oberle, S. Staab, and R. Studer. Ontolift prototype, 2003. http://wonderweb.semanticweb.org.

[30] Z. Xu, X. Cao, Y. Dong, and W. Su. Formal approach and automated tool for translating er schemata into owl ontologies. In H. Dai, R. Srikant, and C. Zhang, editors, *PAKDD*, volume 3056 of *Lecture Notes in Computer Science*, pages 464–475. Springer, 2004.