Context-aware Applications with Distributed Ontologies

Flavio De Paoli and Marco Loregian

Dipartimento di Informatica, Sistemistica e Comunicazione,
Università degli Studi di Milano Bicocca,
via Bicocca degli Arcimboldi 8,
20126 Milano (Italy),
{loregian,depaoli}@disco.unimib.it

Abstract. This paper presents a framework for context-aware applications, with a particular focus on collaboration and pervasiveness. The framework relies on distributed ontologies, which are shared components spread over a network. The architecture of the framework provides for the coexistence of devices with different nature and computing capabilities. The framework implements a peer-to-peer model, encompassing three kinds of peer: ontology-management, context-management, and interaction-management peers. The problem of supporting co-operative work practices in heterogeneous, pervasive computing environments is tackled, in particular with respect to the case of hospital settings.

1 Introduction

Context awareness is a central topic for many research communities. In a general sense, *context* information concerns "the location, identity and state of people, groups and computational and physical objects" [1]. The definition given by Abowd et al. focuses on the relation between users and applications. This is a particularly relevant aspect in pervasive computing environments where actors (either human or technological) mutually influence their behaviors. In other research areas, such as computer supported cooperative work (CSCW) and human-computer interaction (HCI), the notion of context includes other dimensions, such as user's perceptions and experience [2].

Context-aware systems are requested to *extract*, *interpret* and use context information to *adapt* their behavior. The first two topics, contextual knowledge extraction and interpretation, are highly regarded in logics, knowledge representation (KR) and artificial intelligence (AI) research, whereas adaptive systems are widely treated in the areas of knowledge management (KM), distributed systems engineering (DSE), multi-agent systems (MAS), pervasive and ubiquitous computing. This work mostly focuses on the last issue (adaptation), with minor references to the first two.

Moreover, the problem of context awareness can refer to different levels of abstraction, from adaptation of system configuration to device characteristics (i.e., presence of other entities and their characteristics), to access control to knowledge bases according to environmental features and occurring events. At every level, interoperability among actors requires a shared perception of the environment: ontologies provide the necessary elements to describe environments and contexts in human-readable and machine-processable fashion [3]. Ontologies are acknowledged to be useful and powerful tools in knowledge management systems, while their application in lightweight pervasive applications is still matter for advanced research.

The CADO (*Context-aware Applications with Distributed Ontologies*) framework has been designed with a twofold aim: *(i)* to support dynamic configuration of pervasive computing environments (in a lightweight fashion) and *(ii)* promote rich, semantic-enriched interaction for knowledge sharing in collaborative systems.

CADO relies on distributed ontologies (Section 2) that are shared and managed in a peer-to-peer fashion. The goal is to provide for a comprehensive and rich middleware to address collaborative multi-channel applications whose participants can join and leave dynamically, maintain and share their ontologies and occasionally form clusters to establish a conversation. The architecture of the framework is designed to cope with pervasive computing environments that are characterized by devices with different nature and computing capabilities: for example, we are testing the use of digital pens, which are simple input devices, as well as large screens, that allow for richer interaction, tablet PCs, which are powerful portable devices, mobile phones, and so on. The framework is composed of three layers and designed to support mobility of workers in complex work settings. The three layers ensure semantic, context and application interoperability.

The next section presents the motivation of this work, rooting in earlier and ongoing projects, and the definition of a model of the problem. Section 3 explains with a simple scenario how the framework can be applied. Section 4 describes in details the CADO framework for pervasive computing. Section 5 provides details about knowledge sharing, and ontology merging in particular, that is the mechanisms upon which the framework relies. Some related works are referred in Section 6 and then some concluding remarks are given.

2 Background and motivation

The work presented in this paper has been developed within the SWIRLS project ("Supporting Wards with Interactive Resources and Logic-based Systems", [4]), and relying on ATELIER and MILK projects experience.

The SWIRLS project started in 2004 in collaboration with an Italian hospital, aiming at supporting hospital practitioners by means of digitally enhanced artifacts without disrupting habitual work practices. Hospitals consist of several different work sites (e.g., inpatient rooms, laboratories) that are grouped in departments according to organizational constraints and specific practices. The analyzed environment is also characterized by a high degree of mobility and heterogeneity of actors.

Mobility in clinical work has been thoroughly analyzed by Bardram and Bossen [5]: from an "infrastructural" perspective, mobility is driven by the necessity of being in different physical places, e.g., to visit inpatients in their rooms and use specific equipments in laboratories. Workers need to access shared resources, which can be either permanently located in specific rooms or can be moved towards the patients. From a social perspective, people move to meet other people that retain useful knowledge to accomplish a certain task. People need to contact specific persons, for example specialists in specific fields. As a consequence, each actor plays different roles (providing or looking for knowledge) in different situations and different places. For these reasons, digital devices that surround people and are carried along by workers should exhibit different behaviors according to the current situation and, if they are used to access some content, information they present should be modulated according to the context [4], for example according to privacy policies (Section 3).

Experiments in this field have been carried on in the previous project ATELIER ("Architecture and Technologies for Inspirational Learning Environments" [6]). The goal of ATELIER was to enhance the learning experience of architecture and interaction design students. The ATELIER setting was a single room, where a group of students collaborated on the development of a project (e.g., the design of a building or an artifact), and the adopted software infrastructure was centralized. The main problem was to enact flexible manipulation policies on a central ontology collecting the knowledge retained by a whole class of students. In the SWIRLS project some similar manipulation of shared knowledge is needed, in the form of documents or personal information to be accessed and shared: it happens in a larger setting, more dynamic and technologically heterogeneous. A distributed infrastructure encompassing mechanisms to quickly enable the set-up of local contexts (described by ontologies) is required.

In the MILK project ("Multimedia Interaction for learning and Knowing", [7]), a centralized framework supporting interactions within three different environments (office, by means of PCs, social, by means of large interactive screens, and mobile, by means of cellular phones) has been developed. Ontologies are exploited to capture different kinds of knowledge (e.g., domain, organizational) to support semantic document management and supply richer presentation and navigation solutions. Information on people, communities, projects, and documents is visualized and organized by means of semantic links according to the current situation. Multi-channel communication has been supported for communities. The CADO framework, exploiting the experience gained in the MILK project, represents a further step towards the support of larger, and more dispersed workgroups.

In these projects, ontology dynamics is driven by innovation and discoveries, but while processes of conflict solving and consensus reaching over a shared resource are enacted quite naturally in centralized environments, the propagation of knowledge (i.e., changes in the conceptualization) in a widespread distributed environment requires tailored strategies and models for reaching shared agreements.

A key point is that CADO is targeted to support organizations and it does not aim at forming a global consensus. Alignment is a continuous process of mutual agreement in response to divergence (of information, opinions, and practices) among participants. In other words, in the SWIRLS project, by means of a CADO-based infrastructure, context and knowledge representations are local and mutually aligned for groups of actors interacting in a peer-to-peer fashion. From this process stems the reference ontology for each group, community, and environment.

CADO users and components rely on ontologies covering different aspects of the domains of interest, and such aspects can sometimes be disjointed (e.g., technological components of the system, desired behaviors and user expertise).

A **topic ontology** is the part of an ontology that covers a particular aspect of the domain, collecting a set of concepts connected with a *is-a* relation and with other domain-specific relations (later referred to as *intra-topic relations*). Since knowledge is manifold, users exploit different topic ontologies to cover their entire knowledge and expertise. A collective vision over a topic within a group of individuals, such as a work team or a community, can only be achieved by merging (different) topic ontologies. Ontology merging is the "creation of [a] single coherent ontology that includes the information from all the sources" [8]. In CADO only *virtual* merging is employed, in that no physical ontology is produced by merging topic ontologies, but only a temporary *view* over different sources seen as a whole.

A **distributed ontology** can be built from individual representations: "[a distributed] ontology is divided into several component ontologies and [. . .] each ontology [is constructed] individually (perhaps in parallel) by different developers in a distributed environment" [9]. In other words, distributed ontologies are modular and composed of peer components spread over a network.

In the following section a simple scenario is used to explain how the framework can be applied and to which extent knowledge sharing is facilitated. Ontologies are used to describe both human (domain) and technological components, as well as some description of desired behaviors and objectives, expressed in terms of operational constraints. Concepts and relations are emphasized using a different text style.

3 Scenario

In a healthcare scenario, when physicians gather in a room or in a laboratory to discuss and work together their individual knowledge has to be integrated for the scope of the meeting. This means that each of them concurs in the creation of a broader knowledge base with the knowledge stored in local (e.g., laptops) and remote resources (e.g., digital libraries, DBs, etc.). This knowledge base is then accessed by the various devices available to users: either personal, such as PDAs, or shared, such as wall-mounted displays.

In such a case, since actors refer to the same domain (i.e., healthcare), it is reasonable to assume that there is a substantial lexical and conceptual overlapping between actors, that can form a common background to support interaction, that makes the solution of ambiguities simpler [8, 10]. This assumption has been exploited in CADO to facilitate semantic interoperability (Section 5.1).

The shared knowledge base is composed of the documents every participant wants to share and the relations defined in a common ontology that represents the merge of the expertise of the group members. Documents can be passed along participants and personal ontologies can be enriched by collecting information from other participants to facilitate human-to-human interaction and enrich the sharing experience by supporting knowledge discovery, at least for the span of a conversation.

From a technological perspective, the integration of devices with different characteristics is achieved by referring to a common model of the context that also encompasses domain-dependent behavioral rules. Device capabilities and relations with other elements (devices, documents, users, etc.) are included in a specific topic ontology. Other implicit relations emerge by merging ontology modules and by checking explicit domain constraints (axioms, Fig. 2).

The context of the following example is given by a `SharedRoom` hosting some `Device`s. The scenario involves ontologies of different topics: technological aspects, organizational roles and structures, type of documents and medical contents.

In a `SharedRoom` of an `Hospital`, where medical `Consultation`s usually take place, there are different kinds of `Screen`s, some are large and public (e.g., `Whiteboard`s), others are small and private (e.g., `PDA`s). `Employee`s, `Doctor`s or `Nurse`s, are sensed by `RFID` technology to discover when they are in the room. `Document`s can be displayed on shared `Whiteboard`s, but if two or more people are present in the room, only public documents (e.g., `Notice`s or `Article`s) can be displayed while personal Documents (e.g., `Email`s, personal `Note`s) have to be hidden from others' sight.

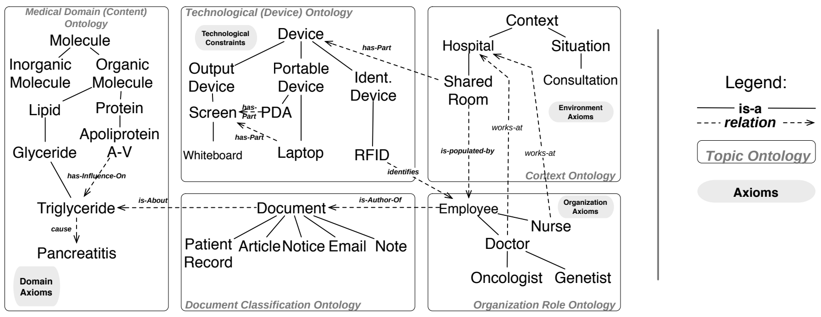


Fig. 1. The (simplified) distributed ontology underlying the example and showing the relations between concepts.

Dealing with ontologies, attributes and relations associated with concepts and axioms supply the context manager with information on specific situations. So for example, a `Document` has a boolean attribute (slot) that states whether a document is public or private, and `RFID` *has-Influence-On* `Employee`, since the `RFID` carried with a badge influences the behavior of the sensed employee. Axioms define the system constraints. For example, the constraint in Fig. 2 states that only public documents can be displayed on large screens if more than one person is present in the room.

A deeper insight of the interaction and reasoning mechanisms can be achieved by the following scenario, which is a realistic example even if not referred to any real hospital situation.

> Dr. Dorian, an `Hematologist`, is evaluating the anomalous production of `Lipid`s in the blood of a patient, but he is not a specialist on the particular subject of such diseases. He is then reading on the `Whiteboard` some `Article`s, checking the `PatientRecord` and jotting some `Note`s on the screen about how to interpret the data. Not being able to solve the problem, Dr. Dorian calls for help Dr. Reid, a senior `Genetist` who has just participated in the last International Conference on Very Rare Genetic Diseases. When

```
(defrange ?screen  :FRAME Whiteboard)
(defrange ?room    :FRAME SharedRoom)
(defrange ?doc     :FRAME Document)

(forall ?room
  (forall ?screen
    (=>
      (and
         (Screens ?room ?screen)
         (>(number-of-slot-values People ?room) 1))
      (forall ?doc
         (and
            (Display ?screen ?doc)
            (not(private ?doc))
 )))))
```

Fig. 2. Sample axiom, written in Protégé Axiom Language (PAL) stating that when more than one person is in the room, a whiteboard, which is a particular wall-mounted screen, should not display personal documents. Private documents have an attribute called *private* set to *true*.

Dr. Reid enters the `SharedRoom` with her own `RFID` badge and `Laptop`, the system detects the new presence and hides from the `Whiteboard` the `PatientRecord` and the private `Notes` of the `Hematologist`, to prevent Dr. Reid from reading them. This situation has been recognized by the system as a `Consultation`, as the result of an axiom in the ontology that defines `Consultation` as a situation with two or more `Doctors` nearby a `Screen` in a `SharedRoom`. The `Documents` stored in the `Genetist`'s `Laptop` are organized according to a highly specialized ontology on the topic, including detailed information (i.e., a very fine-grained taxonomy and several specific relations among concepts) on genetic diseases and of the kinds of involved `Molecules`. Moreover, Dr. Reid owns an electronic version of the proceedings of the conference she attended. The partial knowledge (ontology) Dr. Dorian was relying on can be now completed with the richer ontology brought in by Dr. Reid. The `Whiteboard` then displays `Documents` related to the `Articles` Dr. Dorian was reading as a relation between the two doctors' knowledge is created. For example, a *has-Influence-On* relation between `Apoliprotein_A-V` and `Triglyceride` is discovered in Dr. Reid's ontology (and later propagated to Dr. Dorian's). When a diagnosis is accomplished, after the two practitioners have discussed the patient's anamnesis, Dr. Reid leaves the room and Dorian's personal `Notes` re-appear on the `Whiteboard`. As a result of the interaction, Dr. Dorian's ontology is now enriched with the relations they discussed.

4 The CADO architecture

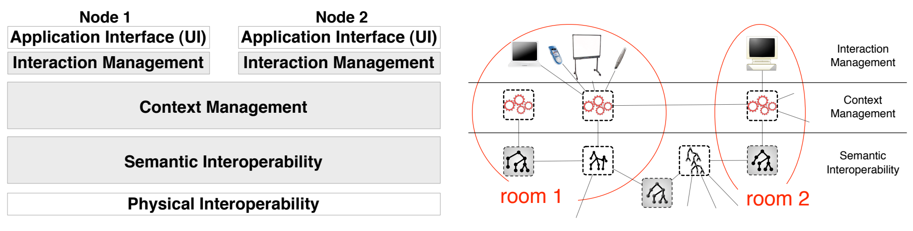
To support the described scenario a framework encompassing various kinds of functionalities is required. A flexible and scalable peer-to-peer architecture allowing for the dynamic discovery of the available components and for the quick reconfiguration of the environment is the most suitable to the model.

In CADO terminology, a **peer** is a software component that can be connected and cooperate with other peers, to form a **cluster**; a **node** is a set of peers that are running on the same hardware. A cluster of nodes may be defined by analogy as a set of communicating nodes, that are connected by some of their peers by means of a **network**.

Peers initially form clusters by means of "physical visibility" (e.g., devices in WiFi range). Only peers that have the possibility to connect with one another may share a perception of the same environment and, therefore, define a common context. At the same time, different contexts can co-exist in the same environment, especially if the connection possibilities are manifold. For example two discussions regarding different topics may occur in the same laboratory, involving different groups of people and different devices. Different context-management services might be needed in such a situation, even in the same local network. On the other hand, this solution does not exclude the possibility of defining a common objective at the application level: a "virtual room" can be created to let people closely interoperate even if they do not share the same physical location.

Fig. 3(a) outlines the CADO architecture layers by showing two nodes. *Physical* interoperability layer lets nodes communicate by addressing issues such as message dispatching, fault tolerance, user authentication. *Semantic* interoperability layer lets user agents process data, understanding their meaning by providing an integrated view over the different ontologies. The *context* interoperability layer lets user agents select data and behavior according to specific situations. The context is defined by the technological and social environment, and application purposes. Every time a cluster with a shared context is formed, a context manager is locally delegated to coordinate the activities of the participants. The *interaction* layer handles interaction with user interfaces. Interaction managers select the base elements to define the common context, acting according to internal (default) logics or enabling user direct interaction.

Fig. 3(b) shows, with a sample situation similar to that described in Section 3, connected peers and involved layers. The interaction layer is composed of various devices (some with high computing capabilities, like PCs and laptops, and with low capabilities, like digital pens and mobile phones). Interaction managers are connected to context-management peers, which may communicate to each other. Ontology-management peers are of two different kinds: adapters and managers. Oval lines indicate clusters of peers forming (virtual) rooms. In room 1 four devices (a laptop, a mobile phone, a Smart


Node 1
Application Interface (UI)
Interaction Management
Node 2
Application Interface (UI)
Interaction Management
Context Management
Semantic Interoperability
Physical Interoperability
Interaction Management
Context Management
Semantic Interoperability
room 1
room 2

(a) (b)

Fig. 3. The layers of the architecture, and an example of setup.

whiteboard, and a digital pen) share the same context and hence are coordinated by the same context manager. This could be the situation in a meeting room where workers, carrying some personal devices, are discussing in front of a whiteboard. Room 2 is an example of virtual clusters among remote peers.

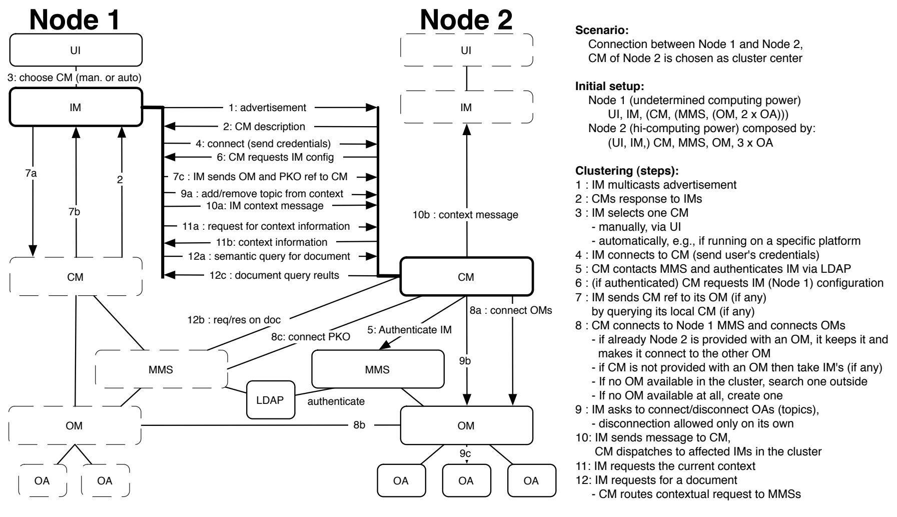
Interaction Managers (IM) are user agents with application-specific or device-specific capabilities that provide users with interaction features and perform cluster discovery and set up. An IM controls the user interface layout and the content information displayed according to context information supplied by context managers. In a (virtual) room cluster, IMs behave according to technological perspective and a social perspective. For example, users in room 1 (Fig. 3(b)) can participate in a face-to-face meeting exploiting the whiteboard that acts as a shared display. Connection and cluster capabilities are constrained by device capabilities. Devices with advanced connectivity support and high performance computing power (e.g., PCs, laptops) will take care of simpler devices (e.g., sensors, RFID readers, digital pens, interactive screens) by driving their behavior to supply users with richer interactive environments.

Context Managers (CM) enact clustering mechanisms and define specific "policies" to drive the behavior of the participants. In the example of Fig. 3(b), the four devices in room 1 are coordinated by the central CM according to its perception of the environment. CMs are contacted by IMs when they fall in range according to casual discovery protocol or when two or more persons decide to cluster. IMs and CMs communicate to each other to establish the *context* for the cluster, and elect a coordinating CM that is in charge of routing information among participating IMs.

Context managers can accomplish their tasks thanks to a set of topic ontologies that describe the different aspects to deal with. In principle, topic ontologies can cover the definition of concepts and relations among them in any domain. In CADO, we mainly deal with the classification of devices and their features, the definition of user roles and preferences, the identification of situations, and finally the reference domain description (e.g., fig. 1). Ontologies are independent resources that are actively published (they are not just mere chunks of information that can be downloaded and edited as it often occurs) and are remotely accessible.

Two kinds of ontologies are foreseen: normative and organization-wide ontologies that are unmodifiable by regular users, and personal ontologies that are selected and evolve according to users preferences and will. The former are often huge and complex, the latter are often simple taxonomies (with a few other relations) that grow over time. In an organization, CADO allows for personal ontology alignment, so to reach a common and shared understanding.

In CADO, two kinds of peers handle ontologies: **Ontology Adapters** (OAs) and **Ontology Managers** (OMs). OAs are in charge of managing a single ontology and ensuring access to it. OMs have the task of merging ontologies to supply their clients (i.e., context managers and ontology managers) with comprehensive (integrated) view and navigation capabilities. These features are at the core of CADO, since, for example, they allow context managers to process the device descriptions supplied by interaction managers and set up a technological profile of the clustering peers.



Scenario:
Connection between Node 1 and Node 2,
CM of Node 2 is chosen as cluster center

Initial setup:
Node 1 (undetermined computing power)
UI, IM, (CM, (MMS, (OM, 2 x OA)))
Node 2 (hi-computing power) composed by:
(UI, IM,) CM, MMS, OM, 3 x OA

Clustering (steps):
1 : IM multicasts advertisement
2 : CMs response to IMs
3 : IM selects one CM
- manually, via UI
- automatically, e.g., if running on a specific platform
4 : IM connects to CM (send user's credentials)
5 : CM contacts MMS and authenticates IM via LDAP
6 : (if authenticated) CM requests IM (Node 1) configuration
7 : IM sends CM ref to its OM (if any)
by querying its local CM (if any)
8 : CM connects to Node 1 MMS and connects OMs
- if already Node 2 is provided with an OM, it keeps it and makes it connect to the other OM
- if CM is not provided with an OM then take IM's (if any)
- If no OM available in the cluster, search one outside
- If no OM available at all, create one
9 : IM asks to connect/disconnect OAs (topics),
- disconnection allowed only on its own
10: IM sends message to CM,
CM dispatches to affected IMs in the cluster
11: IM requests the current context
12: IM requests for a document
- CM routes contextual request to MMSs

Fig. 4. Component communication protocol.

The set of CADO peers is completed by additional peers that supply specific services to users such as Metadata-Management-System (MMS [7]) peers, and Lightweight-Directory-Access-Protocol (LDAP) peers.

4.1 Current implementation

A prototype of the CADO framework has been implemented in Java^{TM} and relies on JXTA^{TM} technology. The JXTA protocol is *network agnostic*, meaning that no assumption is made regarding the network underlying the application. Therefore, it is well-suited to support different networks, including wi-fi, bluetooth and wired LAN.

System components are aware of each other as a result of JXTA discovery mechanisms that allow peers to look for other peers providing the needed service. Peer types (i.e., IM, OM, CM, OA) are visible and recognizable by explicit XML advertisements that express the services each peer offers. CMs are connected with each other as to build a widespread CADO infrastructure, as for ultrapeer nodes in Gnutella networks, upon which peers can form articulated clusters.

Peer clustering is managed by exploiting the JXTA peer-grouping mechanism. IMs connect to CMs to share a view of a certain context with other IMs, by applying knowledge representation published by an OM, around which OAs form another cluster for *semantic interoperability*. Clustering occurs according to specific requirements, e.g., application behavior, client identity (authentication), common interests. To explain the details, consider the typical interaction session in the configuration phase described in Fig. 4. The example shows a node with advanced computational capabilities (e.g., laptop) that enters a cluster managed by a context manager running in another node. When a node wants to participate in a CADO network (e.g., node1 "meeting" node2), the configuration steps are the following:

1-2. The new-comer IM (node1) sends (multicast) advertisement over JXTA network, CMs (that of node2 as well as other CM peers reachable over the network) reply with a message containing descriptions of the services they can provide (including information on IMs already connected and OMs upon which they are relying).

3. The IM selects a CM according to application policies (or under explicit direction of the user). An IM can be instructed to follow criteria such as connect to any CM available, connect to a CM providing a certain service, connect to a CM running on specific platforms. Assume that CM of node2 is elected as coordinator.

4-7. The configuration process occurs. The IM connects to the selected CM and sends user credentials. The CM authenticates the user via LDAP, requests the IM configuration to collect information on the CM in use, exploited OMs and OAs.

8-9. The connections to support semantic interoperability are established. The elected CM selects and connect to the OMs according to its policies. Polices can be enforced by the application or included in the CM configuration. IMs controls the configuration phase by adding/removing specific topic ontologies. Moreover, the CM connects to the participating MMSs.

10-12. The system is configured and the applications can start interacting. IMs can send messages to a CM to:
   1. provide context information: the CM collects it to compute the current context;
   2. request the current context: the CM answers with updated and detailed information on the cluster;
   3. search documents according to semantic queries: the CM will manage the request to provide for access to connected MMSs.

5 Building semantic interoperability

In order to provide context-aware services to IMs, CMs have to manipulate a unified view of the environment as supplied by OMs. In fact the role of OMs is to merge the pieces of ontology (topics) that are exposed by reachable peers (OAs and other OMs). OAs play the role of wrappers for ontology modules, directly attached as local resources, i.e., ontologies expressed in a particular language, or created by a particular software tool. OAs have the task of ensuring interoperability by exposing ontologies in a homogeneous format.

OAs are in charge of managing local ontology evolution according to user-defined policies or default policies that cover the most common situations. Default policies are:

– **Normative policy**: an ontology can only be modified by authorized (usually human) administrators. This policy is usually adopted for reference ontologies (e.g., an organization's ontology) that are maintained off-line.
– **Selective policy**: an ontology can be modified on external requests (from OMs), but only by authorized users and possibly with explicit authorization from the owner. This policy is adopted to deal with sensitive ontologies that reflect user's knowledge representation. The owner might be prompted before a change can take effect.
– **Plastic policy**: an ontology can be freely modified by OMs. This policy is adopted by flexible nodes that need to reflect external events. For example an ontology related to the social interactions should adopt such a policy.

Consider Fig. 5 that shows a sample situation resembling the scenario in Section 3: a physician attended a conference where a paper proposing a new treatment for a pathology was proposed. Her personal ontology was updated to encompass the findings. Later, she attends a meeting and decides to share her personal notes with other physicians. As a result, another physician that was unaware of the new finding, is prompted to update her knowledge base with the new information. This is due to the integration process occurring when the two personal ontologies get in touch by means of an OM.

5.1 Merging of ontologies

Given two (or more) ontologies to be merged, corresponding concepts in the taxonomies of the ontologies have to be found: this requires a similarity measure [11–14] for taxonomies (i.e., a matching criterion). When similarities are found, corresponding concepts have to be merged preserving relations — taxonomy *is-a*, concept-specific attributes, … — and origins have to be tracked, in order to remember to which ontology concepts and relations belong.

In summary, according to literature (Section 6), ontology merging consists of *(1)* finding places in the ontologies where they semantically overlap — either extensionally, i.e., concepts with the same label or synonyms, or intensionally, i.e., concepts with the same place in the taxonomy — *(2)* check the consistency, coherency and non-redundancy of the result. Iteration of such steps might be required.

In the current implementation, the default approach starts with the merging of topic ontologies. For every topic ontology the activities are: *(1)* visit the taxonomy to identify overlapping concepts; *(2)* build the new taxonomy including all the concepts in the original taxonomies; *(3)* add any other *intra-topic* relation to deliver the merged topic ontology. The ontology merging is completed by adding *inter-topic* relations according to the original ontologies. During the merging, simple checks are performed to avoid cycles and replications. Note that tackling the merging problems by considering topic ontologies ensure an higher degree of confidence, than considering generic taxonomies. CADO supports merging customization by overriding similarity evaluation and check procedures with more sophisticated matching criteria (e.g., translations, synonym analysis, WordNet-mediated mapping).

To illustrate the mechanism with an example, consider the merging of a new module (OA), as in Fig. 5. Fig. 5.1 shows the initial situation, with three modules (*b*, *c*, and *d*) already merged, and *a* that is still outside the cluster. When the ontology *a* enters in the

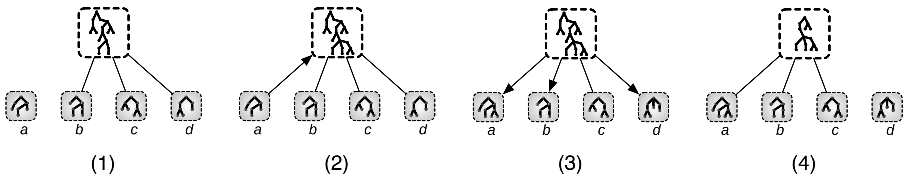


Fig. 5. Example of ontology merging taking place in four steps. Straight lines represent physical connections between components; arrows show the direction of messages being exchanged.

cluster of ontologies coordinated by an OM (Fig. 5.2, and Fig. 4 step 9), the merging process starts by identifying homonyms and merging them in the OM's ontology in a concept containing the relations of both versions.

Through the merging of concepts and exploitation of relations, the new ontology can be merged with the existing one by adding the new discovered links. However, since the ontology exposed by the OM is only a "virtual" merging of the underlying ones (we could even say that it is an alignment more than a merging [8]), the modules (OAs) are notified with the newly discovered connections to let them learn new concepts and relations (Fig. 5.3). Each OA can, in response to the notification, autonomously decide whether to adopt or reject the change. As a result, every module that has accepted the proposed changes is complete and sound, since all relations and axioms refer to known concepts.

In the example, OAs *b*, *c*, and *d* exhibit different behaviors in response to the arrival of *a*. Ontology *d* is affected and changed by the alignment with *a*, that is intuitively depicted by a graphical change in the taxonomy. If *a* is the ontology adopted by a certain user in a shared environment to index some contents, *d* may be the ontology adopted by another user in a shared environment. Module *c* is not affected by the arrival of *a*, it might be the case of the ontology of another user with different interests than *a*'s, so that there is no overlapping or relation between their ontologies. Module *b* could be affected by the knowledge retained in *a*, and that is depicted by the arrow coming from the OM, but it is not accepting any change. This might be the case of a node with a normative role in the organization, such as an enterprise server or a shared standard categorization. Note that *a* is not only supplying new information, but it is also receiving some, as a result of the alignment with other modules. When an ontology module leaves a cluster (e.g., *d* in Fig. 5.4), the description of the part of the context it represents has to be removed from the "virtual" ontology managed by the OM. Anyway, concepts and references that were accepted (i.e., learned) by the other modules still exist in the global ontology. Moreover, the departing module will bring around the acquired knowledge to "contaminate" other peers in the system. In such a way, the knowledge diffusion and long-term convergence is ensured.

5.2 Knowledge sharing

The CADO framework provides the facilities to create networks to share and exchange knowledge among actors in heterogeneous environments, by means of fixed and mobile devices.

The process of knowledge transfer can take place at different levels: between individuals, from individuals to groups, between groups, and from group to the organization [15]. Depending on the kind of knowledge transfer, pieces of information can gain different importance. The problem of exploiting "hidden" knowledge, which is not public and is retained by individuals in their own repositories, if not in their minds (tacit knowledge), can only be overcome by fostering communication, collaboration, and thus knowledge sharing. Knowledge networks are constituted by individuals with similar interests, or that have the need to cooperate to pursue a common goal.

Sharing knowledge means more than providing shared access to information, since learning and expertise transfer have also to be fostered. Even if knowledge networks system designers are aware of the issue, very often those same systems fail because they require additional effort to users and also lack in pervasiveness and proactivity (i.e., knowledge discovery functionalities). The additional amount of time each user has to spend to nurture his/her knowledge network is very often the first cause of "death", or dismantlement for knowledge network systems. Aim of the CADO framework is to enable the creation of systems that do not require such an extra effort. Applications designed, and implemented, in the framework exploit contextual information to complete the perception users have of the environment, by enriching their knowledge bases with relevant information without explicit search operations on focus topics.

Furthermore new knowledge acquired should not be dispersed once a conversation is concluded. Even if the (network) connections dissolve, the achieved consensus should be propagated in space (i.e., by people moving to other locations with new documents stored in their personal devices) and time (i.e., knowledge will persist in the place in a integrated form until some of the devices are connected: local repositories are updated by meeting participants).

In other words, in CADO systems, knowledge propagation occurs via direct exchange when people get in touch and their devices can communicate with each other. Moreover, what has been learned during a conversation can also become persistent at statically located peers (if they act as servers) so to propagate knowledge to other people entering the same location. It is then a knowledge propagation model that works both for static and mobile (nomadic) actors. Similar models have already been presented in literature under the name of epidemic [16] or gossip-style [17] models.

6 Related works

Building ontologies by merging means to find similarities and mappings, and merge concepts, relations, axioms, from existing ontologies to build a new ontology on the same subject [18]. We defined a subject, or a context, as a set of related topics, and described the general steps by which merging can be achieved. General mechanisms have been presented for the merging of ontologies, to extend an existing ontology and keep different versions of an ontology aligned (and to support replication) [19, 20].

A recent proposal presented a machine-learning approach to semantic alignment of ontologies, namely APFEL. The general observation on which APFEL is based is that alignment methods can be mapped onto a *generic process*, that is described with details in [21]. APFEL method aims at improving existing alignment strategies, also by involving users in critical steps (including the selection of an alignment strategy among the existing ones).

Noy and Musen [8] presented PROMPT, a tool that provides semi-automatic facilities for merging and aligning ontologies. The tool is independent from the specific ontology representation (i.e., syntax, language) and is based on Protégé, an open-source ontology editor and knowledge acquisition systems that we have adopted in our experiments. Chimaera [10, 22] is another example of interactive environment for discovering mappings between ontologies, and to merge ontologies. Both PROMPT and Chimaera rely on lexical overlapping of concepts (matching terms). Feature-based approaches have also been presented, meaning that two concepts are considered to be matching if they have common features (functions, parts, and attributes) [23]. Using an approach of this kind, also the degree of matching can be assessed and expressed.

7 Concluding remarks

One of the major problems in pervasive computing environments is the capability of systems to understand the context and to change their behavior accordingly. The framework presented in this paper addresses the issue from two perspectives: clustering together actors to support collaboration via shared interfaces and interactions; and merging shared ontologies to form short-term and long-term consensus. The former perspective is supported by Context Managers, which can understand the context and drive actors' devices accordingly, the latter is supported by Ontology Managers, which provide for dynamic merge of personal knowledge bases.

Knowledge management issues are tackled from different perspectives: *(a) technological*, since dynamic automatic configurability of environments is achieved, *(b) organizational*, since the framework allows the integration of normative knowledge sources with subjective conceptualizations of users, *(c) social*, since it is possible for users with different roles to interact in a facilitated way, *(d) personal*, since it is possible for users to play different roles and yet not lose their identity and knowledge.

Future activities will deal mainly with extensive testing and validation of the proposed solution. In fact, the adoption of new technologies changes the personal and organizational practices and promotes new needs that should be taken into account to propose effective solutions. The use of familiar devices, such as cellular phones, laptops and digital pens, is crucial to address the problem of making people accept innovative solutions, according to pervasive-computing paradigm, that are not disruptive of acquired habits.

Acknowledgements

The work presented in this paper was partially supported by the ATELIER project (Architecture and Technologies for Inspirational Learning Environments, IST-2001-33064) and by the MILK project (Multimedia Interaction for Learning and Knowing, IST-2001-33165).

References

[1] Abowd, G.D., Dey, A.K., Brown, P.J., Davies, N., Smith, M., Steggles, P.: Towards a better understanding of context and context-awareness. In: HUC '99, London, UK, Springer-Verlag (1999) 304–307

[2] Chalmers, M.: A historical view of context. Comput. Supported Coop. Work **13** (2004) 223–247

[3] Gruber, T.R.: A translation approach to portable ontology specifications. Knowl. Acquis. **5** (1993) 199–220

[4] Boselli, R., Cabitza, F., De Paoli, F., Loregian, M.: An Adaptive Middleware to Support Context-Aware Knowledge Sharing. In: ICDCSW '05, IEEE Computer Society (2005) 352–358

[5] Bardram, J., Bossen, C.: Mobility work: The spatial dimension of collaboration at a hospital. Computer Supported Cooperative Work **14** (2005) 131–160

[6] Calegari, S., Loregian, M.: Ontologies help finding inspiration: a practical approach in multimedia information management. In: Proceedings of PAKM2004. Volume 3336 of LNCS., Springer-Verlag (2004) 307–318

[7] Boselli, R., Dondi, R., De Paoli, F.: Knowledge organization and retrieval in the MILK system. In: Proceedings of SEKE 2003. (2003) 372–376

[8] Noy, N.F., Musen, M.A.: Prompt: Algorithm and tool for automated ontology merging and alignment. In: Proceedings of AAAI-2000, AAAI Press / The MIT Press (2000) 450–455

[9] Sunagawa, E., Kozaki, K., Kitamura, Y., Mizoguchi, R.: An environment for distributed ontology development based on dependency management. In: Proceedings of ISWC 2003. Volume 2870 of LNCS., Springer (2003) 453–468

[10] McGuinness, D.L., Fikes, R., Rice, J., Wilder, S.: An environment for merging and testing large ontologies. In: KR2000. (2000) 483–493

[11] Resnik, P.: Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language. Journal of Artificial Intelligence Research **11** (1999) 95–130

[12] Bisson, G.: Learning in fol with a similarity measure. In: AAAI. (1992) 82–87

[13] Maedche, A., Staab, S.: Comparing ontologies— similarity measures and a comparison study. Technical report, Institute AIFB, University of Karlsruhe (2001)

[14] Ehrig, M., Sure, Y.: Ontology mapping - an integrated approach. In Bussler, C., Davies, J., Fensel, D., Studer, R., eds.: ESWS. Volume 3053 of Lecture Notes in Computer Science., Springer (2004) 76–91

[15] Alavi, M., Leidner, D.E.: Review: Knowledge management and knowledge management systems: Conceptual foundations and research issues. MIS Quarterly **25** (2001) 107–136

[16] Gupta, I., Kermarrec, A.M., Ganesh, A.J.: Efficient epidemic-style protocols for reliable and scalable multicast. In: SRDS, IEEE Computer Society (2002) 180–189

[17] Chlebus, B.S., Kowalski, D.R.: Gossiping to reach consensus. In: SPAA '02, New York, NY, USA, ACM Press (2002) 220–229

[18] Ashpole, B., Ehrig, M., Euzenat, J., Stuckenschmidt, H., eds.: Integrating Ontologies '05, Proceedings of the K-CAP 2005 Workshop on Integrating Ontologies, Banff, Canada, October 2, 2005. Volume 156 of CEUR Workshop Proceedings., CEUR-WS.org (2005)

[19] Pinto, H., Prez, A., Martins, J.: Some issues on ontology integration. In: Proceedings of the Workshop on Ontologies and Problem Solving Methods during IJCAI-99. Volume 7. (1999) 1–12

[20] Pinto, H.S., Martins, J.P.: A methodology for ontology integration. In: Proceedings of the international conference on Knowledge capture, ACM Press (2001) 131–138

[21] Ehrig, M., Staab, S., Sure, Y.: Bootstrapping ontology alignment methods with APFEL. In Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A., eds.: ISWC. Volume 3729 of LNCS., Springer (2005) 186–200

[22] McGuinness, D.L.: Conceptual modeling for distributed ontology environments. In Ganter, B., Mineau, G.W., eds.: ICCS. Volume 1867 of Lecture Notes in Computer Science., Springer (2000) 100–112

[23] Rodríguez, M.A., Egenhofer, M.J.: Determining semantic similarity among entity classes from different ontologies. IEEE Trans. Knowl. Data Eng. **15** (2003) 442–456