# Context information for knowledge reshaping

C. Bolchini, C.A. Curino, E. Quintarelli*, F.A. Schreiber and L. Tanca

Dip. Elettronica e Informazione
Politecnico di Milano
P.zza L. da Vinci, 32
20133 Milano, Italy
E-mail: bolchini@elet.polimi.it
E-mail: curino@elet.polimi.it
E-mail: quintarelli@elet.polimi.it
E-mail: schreiber@elet.polimi.it
E-mail: tanca@elet.polimi.it
*Corresponding author

**Abstract:** More and more often, we face the necessity of extracting appropriately reshaped knowledge from an integrated representation of the information space. Be such a global representation a central database, a global view of several ones or an ontological representation of an information domain, we face the need to define personalised views for the knowledge stakeholders: single users, companies or applications. We propose exploiting the information usage *context* within a methodology for *context-aware data design*, where the notion of context is formally defined together with its role within the process of view building by *information tailoring*. This paper presents our context model, called the *context dimension tree*, which plays a fundamental role in tailoring the information space according to user information needs.

**Keywords:** context model; data tailoring.

**Reference** to this paper should be made as follows: Bolchini, C., Curino, C.A., Quintarelli, E., Schreiber, F.A. and Tanca, L. (2009) 'Context information for knowledge reshaping', *Int. J. Web Engineering and Technology*, Vol. 5, No. 1, pp.88–103.

**Biographical notes:** Cristiana Bolchini (MS, 1993) received her PhD in Automation and Computer Science Engineering in 1997 from Politecnico di Milano, Italy. Since December 2003, she has been an Associate Professor at the Dipartimento di Elettronica e Informazione of the same institution. Her research interests are twofold: they cover both issues related to data management for context-aware, pervasive systems and methodologies for the design of embedded systems with dependability properties. On these topics, she has published about 80 papers on refereed international journals and conference proceedings. She is currently an Associate Editor of *IEEE Transactions on Computers*.

Carlo A. Curino received his Bachelor's degree in Computer Science at Politecnico di Milano. He participated in a joint project between the University of Illinois at Chicago (UIC) and Politecnico di Milano, obtaining a Master's in Computer Science at UIC and a Laurea Specialistica (cum laude) at Politecnico di Milano. During his PhD studies at Politecnico di Milano, he spent over a year as a Visiting Researcher at the University of California, Los Angeles

(UCLA), where he currently has the same position. His recent research interests include context awareness, data integration, schema evolution and temporal databases.

Elisa Quintarelli received her Master's in Computer Science from the University of Verona, Italy. On January 2002, she obtained her PhD in Computer and Automation Engineering at Politecnico di Milano and is now an Assistant Professor at the Dipartimento di Elettronica e Informazione of the same university. Her main research interests concern the study of efficient and flexible techniques for specifying and querying semistructured and temporal data and the application of data-mining techniques to provide intensional query answering. More recently, her research has been concentrated on context-aware data management.

Fabio A. Schreiber is a Full Professor of Databases and Technologies for Information Systems at the Department of Electronic and Information Engineering of the Politecnico di Milano, Italy. He also teaches the Embedded Databases course in the ALaRI Master's programme in Embedded Systems at USI in Lugano (CH). His current research interests include distributed information systems design, database systems for context-aware applications, very small and mobile database design methodologies and applications, data management in pervasive systems and WSNs. On these and other topics, he authored 100 papers published in international journals and conferences. He is a member of the Editorial Board of *Data and Knowledge Engineering*.

Letizia Tanca obtained her PhD in Computer Science in 1988 and is currently with Politecnico di Milano, Italy, as a Full Professor. She teaches courses on Databases, Foundations of Computer Science and Information System Technologies. She is the author of over 100 papers on databases and database theory and the book *Logic Programming and Databases* with S. Ceri and G. Gottlob. She has been the Local Leader of national and international projects and a Referee for international journals and conferences. Her most recent research interests concern semantics-based data integration, context-aware databases and small databases for mobile devices.

## 1 Introduction

Internet access and knowledge sharing is becoming a driving technology to give companies competitive advantages, offer end users innovative services and enhance scientific communication in the research world. However, the vast amount of globally available information risks flooding human users as well as application systems – whether these are devoted to simple data management or sophisticated reasoning tasks. In fact, effective knowledge usage requires that different information resources be appropriately integrated and personalised so that the user (human or machine) is not confused by too much noise and always equipped with an adequate share of knowledge for his/her task and for the operation environment.

To achieve this goal, the state-of-the-art techniques for database design must be extended to consider also users' activities, tasks and intentions (see Jiang *et al.*, 2006; Rajugan *et al.*, 2006; Wouters *et al.*, 2005). However, personalisation is not enough: indeed, user (or application) information needs cannot be merely defined as a profile,

but must also be shaped according to the current time, situation and interests, altogether forming the user's *context*. Context is, thus, the key metaknowledge which must be formally defined and whose role becomes essential within the process of view design; as a consequence, database design must target two different realms: the reality of interest, which is captured by the application domain model (for example, an Entity-Relationship (ER) schema or a domain ontology) and context metaknowledge, which is used to reshape the whole amount of available information on the particular needs that the destination user is experiencing in a particular situation. While classical data models at a conceptual or logical level are perfectly suited to the first target, context modelling and, consequently, the context's relationship with the data, present different challenges and deserve appropriate consideration.

Note that the above two tracks of context-aware data design may be totally independent since no feedback is needed between them until the final phase of the design process, when the context elements are associated with the relevant domain information.

The problem of context modelling was first encountered within the *Context-ADDICT* (Context-Aware Data Design, Integration, Customisation and Tailoring) project (Bolchini *et al.* 2006b; 2004; 2007), aimed at the definition of a complete framework which, starting from a methodology for the early design phases, supports mobile users through the dynamic hooking and integration of information sources as soon as they become available to deliver a *context-based* portion of data to their mobile device. However, while *Context-ADDICT* is specifically targeted to support mobility, thus taking into account such problems as low power, frequent disconnections and resource scarceness, we believe that context-aware view design is useful in itself in the many situations which require a top-down approach, where the information contained in a (virtual or physical) large repository must be shaped *a posteriori*, according to the user's needs.

Today, a lot of research is devoted to context-aware ubiquitous systems, leading to the development of a variety of context models. Interesting surveys on context-aware systems and models have been proposed in Baldauf *et al.* (2004), Chen *et al.* (2003), Kaenampornpan and O'Neill (2004) and Strang and Linnhoff-Popien (2004) and a theoretical framework for defining ontological context-dependent views has been proposed in Rajugan *et al.* (2006) and Wouters *et al.* (2005). However, none of the considered systems specifically and effectively addresses the problem of using a context model for view design; hence, the decision to produce yet another context model, which supports the main issues involved in the view design task.

The paper is organised as follows: in Section 2, we review and classify the models of context currently available in the literature. Sections 3 and 4 present our context model and its formal definition. Section 5 shows an example of the usage of the model for views definition and in Section 6, we highlight the features that we consider essential in a context model for knowledge tailoring and that have been included in ours.

## 2 Context models for knowledge reshaping

Many approaches have been proposed to define the notion of context and several studies have designed and implemented adaptive applications by introducing the notions of user profile and context (Aberer *et al.*, 2004; Agrawal and Wimmers, 2000; Buchholz *et al.*, 2004; De Virgilio and Torlone, 2005; MAIS Project, 2002–2005; Ouksel, 2003;

Rajugan *et al.*, 2006; Torlone and Ciaccia, 2003; Wouters *et al.*, 2005). Although interesting surveys have been proposed in Baldauf *et al.* (2004), Chen and Kotz (2000), Kaenampornpan and O'Neill (2004) and Strang and Linnhoff-Popien (2004), during the first phases of our work, we felt the need for a framework to systematically evaluate context models, defining a set of relevant, objective and rather general categories of model features. The result of this analysis is the conscious inclusion (exclusion) of those features in (from) the Context-ADDICT context model, having in mind the objective of supporting the data tailoring task in a context-aware environment. The features that we isolated and classified are discussed in this section, with a brief explanation.

A first set of features includes all elements that are used to express the *modelled aspects*; in detail, they are:

- *Space* – whether the considered context model includes a location parameter.
- *Time* – whether the considered context model includes a parameter representing time.
- *Absolute/Relative space and time* – whether the space and time parameters (if any) are represented absolutely (*e.g.*, Greenwich Mean Time (GMT) and Global Positioning System (GPS) coordinates) or relatively (*e.g.*, 'near something', 'last month', 'after that').
- *Context history* – whether the history of previous contexts is part of (relevant for) the context itself; the parameter is used to specify if the current context state depends on previous ones or is a pure snapshot of the user's current environment.
- *Subject* – who or what is the subject of the described context. This feature refers to the point of view used to describe the context itself; some models describe the context as perceived by the user, while others assume the application point of view, considering, as a consequence, the user itself as part of the context.
- *User profile* – whether the user profile (in terms of preferences and personal features) is represented in the context model; if this is the case, the parameter expresses the adopted modality (*i.e.*, the system describes the user's characteristics one by one or provides a role-based model of user classes).

The second set of features we grouped refers to the *representation features* that the context models exploit, identified as:

- *Type of formalism* – refers to the class of the conceptual tool used to capture the context (key value, markup, logic, graph, ontology-based). Different classes provide different features (*e.g.*, high or low intuitiveness, possibility to be automatically processed, reasoning support, formal semantics) and are more or less adequate for certain applications.
- *Level of formality* – whether the context model is a formal one and whether the formalisation expresses intuition well.
- *Flexibility* – the model's ability to adapt easily to different contexts: a model can be 'application domain-bounded' if it is substantially focused on a single application or on a specific domain or 'fully general' if it can naturally deal with very different domains or applications (*i.e.*, is it possible to capture any kind of context with this model and how easy is it?).

The last group of features includes all the parameters characterising *context management and usage*, useful to determine how the model can be used and maintained; the set includes:

- *Context construction* – highlights if the context description is built centrally or via a distributed effort; this indicates whether a central (typically design-time) description of the possible contexts is provided or if a set of peers reaches an agreement about the description of the current context at runtime.
- *Context reasoning* – indicates whether the context model enables reasoning on context information to obtain more abstract or more complex context descriptions.
- *Context information quality monitoring* – when the context data are perceived by sensors, the system might explicitly consider and manage the quality of the retrieved context information.
- *Ambiguity and incompleteness management* – in case of ambiguous, incoherent or incomplete context information, it points out if the system is able to 'interpolate' and 'mediate' somehow the context information and reconstruct a plausible 'current context'.
- *Automatic learning features* – highlights whether the system observing the user behaviour or environment is able to automatically deduce knowledge about the user's context; *e.g.*, by studying the user's browsing habits, the system learns user preferences.

This categorisation highlights the focus of the modelling effort, how it is represented and how the context information is exploited: we consider this information fundamental to understand the meaning and value of the context model under scrutiny and we used it while examining the existing context models to decide whether they could be suited for the knowledge reshaping task.

As a first step, we reviewed the existing context models and classified them by identifying the following five classes of usage of context:

1 *Context as a matter of channel-device-presentation* – Systems of this class are characterised by a *variable context granularity*, a low level of formality, limited flexibility (often considering only specific applications) and usually having the *application as the subject* of the model. Furthermore, the management of location and time dimensions is usually limited or absent, *feature-based user profiling* is present and there is a *centrally defined context*. While automatic learning features can be available, context quality monitoring, ambiguity management and context reasoning are generally not supported. In this class, we can include the works of Bolchini *et al.* (2006c), Buchholz *et al.* (2004), CoDaMoS development team (2003), De Virgilio and Torlone (2005), De Virgilio *et al.* (2006), Preuveneers *et al.* (2004), Gu *et al.* (2005) and Kaenampornpan and O'Neill (2004).

2 *Context as a matter of location and environment* – Models of this class in general provide precise *time and space management*, a high degree of flexibility and a *centralised context definition*. Context reasoning may be provided, offering a powerful abstraction mechanism; on the other hand, automatic learning is rarely exploited. *Information quality management* and *disambiguation* may be available, particularly when the context information is acquired by sensors. In this class,

we can include the works of CoDaMoS development team (2003), Preuveneers *et al.* (2004), Fahy and Clarke (2004), Gu *et al.* (2005), Kaenampornpan and O'Neill (2004), Petrelli *et al.* (2000) and Strimpakou *et al.* (2006).

3 *Context as a matter of user activity* – The focus of this class of models is on 'what the user is doing'; consequently, *context history* and *reasoning* are important issues. Time and space are considered relevant as far as they provide information about the user's current activity.[1] While the level of formality may vary, the *context definition* is generally *centralised* and the *user* is the *subject of the model*. When available, *automatic learning* is used to guess user activity from sensor readings. In this class, we can include the works of Kaenampornpan and O'Neill (2004), Petrelli *et al.* (2000), Sridharan *et al.* (2003) and Strimpakou *et al.* (2006).

4 *Context as a matter of agreement and sharing (among groups of peers)* – The approaches of this group focus on the problem of reaching an agreement about a context shared among peers; as a result, the *context definition* is *distributed* and *context reasoning*, *context quality monitoring* and *ambiguity and incompleteness management* are key issues. Sophisticated location, time and user profiling features are seldom available in models of this class. The *level of formality* is rather *high* due to the need to share information. In this class, we can include the works of Chen *et al.* (2004), Kaenampornpan and O'Neill (2004) and Ouksel (2003).

5 *Context as a matter of selecting relevant data, functionalities and services* – The models of this group focus on how the context determines which data, application functionalities and services are relevant. Context definition is typically centralised and context history and reasoning are often not provided; time, space and user profiles are generally highly developed and well-formalised. Flexibility is usually high while automatic learning features, ambiguity management and information quality are not key issues and are seldom available. The key features of this group are: *the application as subject*, the possibility to express both *variable context granularity* and *valid context constraints* and *multicontext models*. In this class, we can include the works of Bolchini *et al.* (2006a), Kaenampornpan and O'Neill (2004), Rajugan *et al.* (2006), Wouters *et al.* (2005) and Yang *et al.* (2006).

At the end of this comparison, we feel that the systems whose aims are to be completely general and support the context modelling problem as a whole for any possible application do not suit the data tailoring requirements. In fact, the practical applicability and usability, although not discussed because they are rather subjective, are important parameters and often inversely proportional to the generality of the model: the more expressive and powerful, the less practical and usable.

Different context subproblems and applications have almost incompatible requirements and common solutions are still not available. As a consequence, the context model should be chosen depending on the target class of applications; the analysis framework that we proposed can, in this sense, be used by an application designer to choose among the available models or define the requirements of a new context model, if necessary.

While reviewing the context models available in the literature, we realised that knowledge tailoring needs a number of features that are present in some models, but not always together. Indeed, models for the tailoring task should focus on features of the cited above class 5.

The relevant features we have identified have thus been used in our research work as the set of requirements for a generic context model for knowledge reshaping. Our approach mainly differs from the theoretical framework proposed in Rajugan *et al.* (2006) because we propose a general model that can be adopted for creating personalised views for any kind of data source. Moreover, we identify the main perspectives/components (*i.e.*, concepts and relationships) of the notion of context.

## 3 Origin and rationale of the context dimension tree

The definition of a suitable context model is the first and foremost step in the realisation of a methodology for context-aware data view design; this section focuses on this fundamental element, from the original formulation to the current one.

The design methodology defined within the *Context-ADDICT* project (Bolchini *et al.*, 2004; 2007), while establishing the principle that context-aware database design needs the definition of a context model, introduced it as an array of *Ambient Dimensions* used to model the perspectives from which the data are viewed and to derive the context as one or more instantiations of such an array.

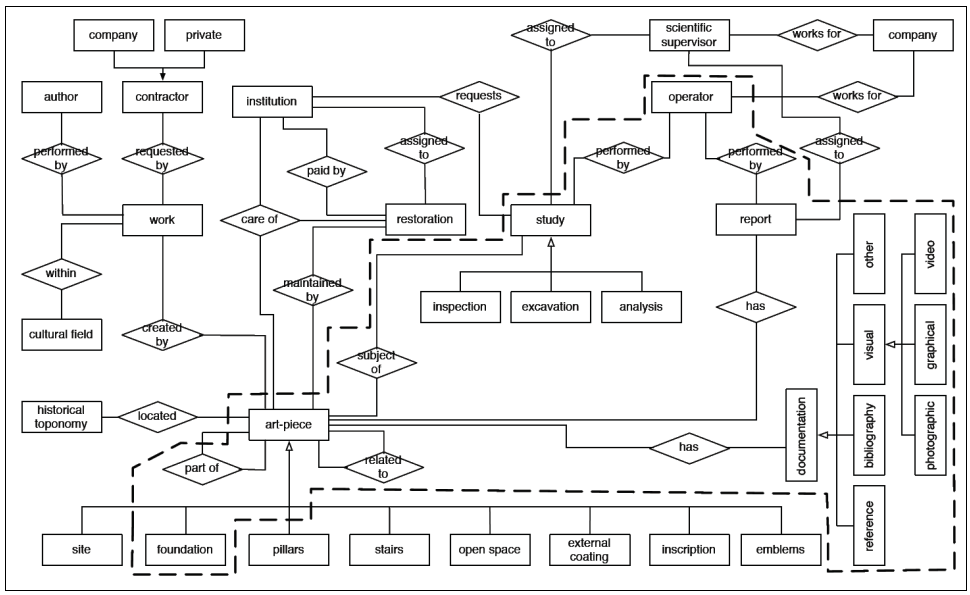
By specifying a value for each ambient dimension, a *chunk configuration* is identified, determining a point in the multidimensional space which represents the context; based on the chunk configuration, a portion of the entire data set, *i.e.*, a chunk, is specified at design time, to be selected later at runtime: personalised user views are composed by one or more such chunks. Dimensions can express the role the data user plays in the application (*role* dimension), what the user's interests are when consulting the data (*interest_topic* dimension), the physical (geographic) area of interest (*space* dimension) and the data life span which is currently of interest (*time* dimension). A further dimension expresses how to use the available data (the *interface* dimension): for instance, the information user might be a human, an application or some other electronic device featuring an embedded system; thus, in the latter cases, storing simple codes may suffice and user-friendly texts are not required. Another dimension takes into account different working phases or modes (the *situation* dimension), allowing one to select only data pertinent to the current phase of the information system life.

Our experience confirms that the dimensions listed above are useful; yet, they may be integrated with new ones when there is an opportunity to better classify and differentiate context features or eliminated whenever the application space does not lend itself to that particular analysis.

Consider now the problem of supporting a context-aware application for the domain of the exploration, study and visit of archaeological sites.

Figure 1 contains the ER schema of the application, describing the archaeological patrimony, maintenance actions, supporting institutions and operational personnel. The dotted line-bordered area represents a context-aware view over the schema and will be discussed later in the paper.

**Figure 1** The global schema for the archaeological site




Note: The elements related to the selected chunk configuration are those inside the dashed line.

The actors and information users in this application are the scientific and administrative supervisors, site operators (such as artwork reporters and surveyors) and tourists. According to their roles, these people may be endowed with different kinds of computation means: the supervisors with their personal computers and the on-site workers and tourists with mobile devices.

Figure 2 lists a set of possible values for each dimension. In principle, the dimension values of Figure 2 could be combined in every possible way, each representing a possible (component of a) context, *i.e.*, a *chunk configuration*; as will be shown later, this is not always the case.

**Figure 2** Ambient array model: an example of the dimensions and possible values

| role | interest_topic | situation | interface | time | space |
|---|---|---|---|---|---|
| operator<br>supervisor<br>visitor | art-pieces<br>employees | routine<br>recovery | human<br>machine | today<br>this-month | this-site<br>this-area |

This is an example of a possible chunk configuration:

*role*=`supervisor`, *interest_topic*=`art-pieces`,
*situation*=`recovery`, *interface*=`human`,
*time*=`this-month`, *space*=`this-area`.

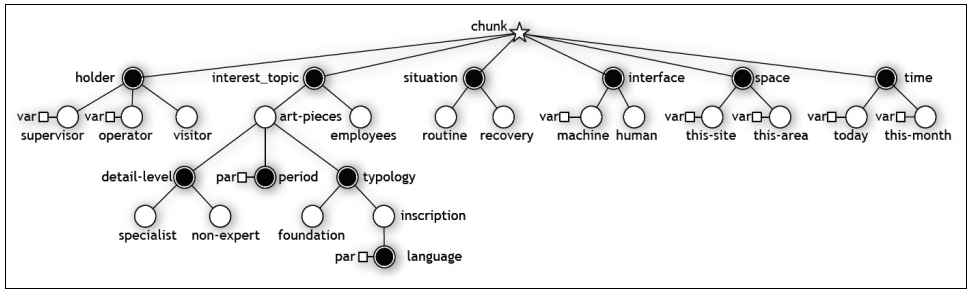
Such a configuration is used to single out the data about the art pieces needed by the site supervisor when performing a recovery action. The data must be human-readable (text or other) and concern the whole area of the current archaeological site listed in the database and the current month.

The *space* and *time* dimensions deserve further discussion: indeed, these dimensions can be conceived either as *a priori*, relative to the current time or position of the user (*i.e.*, based on input from the environment – *e.g.*, GPS or system clock) or as absolute, *i.e.*, obtained by explicit specification. Consider the *space* dimension in Figure 2: here, it is used in the relative sense and its values are `this-site` or `this-area`; this means that, by centring on the user's location, the dimension values only state a granularity or a distance from that. The same holds for the *time* dimension, whose values in the example are `today` and `this-month`; still in a relative sense, we could contemplate values like `next week` or `the last ten days`, but again the origin point would be the current time. In absolute time semantics, values for the time dimension could be `January` or `year 2004`.

It is worth noting that dimensions are not always orthogonal; for example, in our application, the *time* dimension is not significant for all the considered roles. In particular, for the `visitor` role, its specification is irrelevant, whereas it is very significant in the planning activity of the `supervisor` and the `operator` roles.

The ambient array model can express interesting information about the context and present the different dimensions; nevertheless, it is limited as far as dimension values are concerned due to the fact that a flat, one-level view is adopted. This may suffice for some of the dimensions (such as, for instance, the *situation* and *interface* dimensions), but in a more general case, a further refinement through a hierarchy of values (such as the case of *interest_topic*) is in order. As an example, let us consider in the archaeological application domain the concept of `art-pieces`. Here, the user might be interested in different typologies of historical data (*e.g.*, `foundations` rather than `inscriptions`) or different levels of detail in the information (*e.g.*, `specialised` or `non-expert`). The model has thus been extended to a hierarchical structure, the context dimension tree, whose application in the running example is shown in Figure 3.

**Figure 3** The context dimension tree for the running example on archaeological sites



## 4 Definition of the context dimension tree

Let us use the example of Figure 3 to understand the structure of a context dimension tree. The context dimension tree is a tree $T = \langle N, E, r \rangle$ satisfying the following properties:

- $N = N_D \cup N_C \cup N_A$ such that $N_D$, $N_C$ and $N_A$ are pairwise disjoint. $N_D$ represents the set of *dimensions*, *NC* the set of *concepts* and $N_A$ the set of *attributes*. Let us momentarily postpone the explanation of *attribute nodes*. The two main types of nodes are: *dimension nodes* (*ND*, black) and *concept nodes* (*NC*, white). The first-level nodes are dimension nodes, the ambient dimensions we have already talked about; the children of a dimension node are concepts that express the values for such dimension. Concept nodes can be, in turn, further specialised, allowing a more refined tailoring of the data with respect to that perspective. Indeed, a concept may be characterised by different aspects which explain why the corresponding node becomes the parent of new subdimensions. Consider Figure 3: while the children of the *situation* dimension are the values already considered in Figure 2, the *interest_topic* value art-pieces is subject to further refinement: it can be analysed in terms of detail-level, period and typology, which become the new subdimensions admitting new values. Note that the process may obviously be iterated any number of times and it is a design tradeoff choosing the right level of detail.
- $r \in N_C$, *i.e., the root of the tree is a concept node.* The root of the tree models all the possible contexts; thus, it represents the entire dataset before tailoring.
- The edge set is composed by two disjoint subsets: $E = E_R \cup E_A$, such that $E_R \cap E_A = \emptyset$. $E_R$ represents the *subelement of* relationships, whereas $E_A$ represents the *attribute of* relationships.
- $\forall e = \langle n, m \rangle \in E_R$, either $n \in N_D \wedge m \in N_C$ or $n \in N_C \wedge m \in N_D$; *i.e., a dimension node has concept nodes as children and a concept node has dimension nodes as children.* Coherent with the meaning of dimension and concept nodes, each 'generation' contains nodes of the same colour and colours are alternated while descending the tree.
- We now consider *attribute nodes*. $\forall n \in N_A$, $\forall e = \langle m, n \rangle \in E_A$, $m \in N_D \cup N_C$; *i.e.*, the father of an attribute node is either a dimension node or a concept node. In the first case, the attribute represents a *parameter*, while in the second case, it represents a *variable*. Consider the subdimensions of `art-pieces`: while `detail-level` and `typology` have white children (*i.e.*, their values), `period` only features a small square, an *attribute node*; this attribute is a selection parameter whose instances represent the possible values of the `period` dimension, *e.g.*, the century when the art piece was created, like `VI b.c.`,`V b.c.`. Thus, dimension branching can be represented in both ways: either by explicitly drawing the values as white nodes or by indicating a *parameter* whose instances are the dimension values. This does not modify the expressive power of the model: rather, it makes it more readable and more usable for the designer.

  On the other hand, white nodes may feature a *variable* indicating how to select a specific set of data instances. As an example, consider the variable associated with the `supervisor` role: it is an identifier used, at runtime, to select exactly the data related to a specific supervisor.

- $\forall n \in N_A, \neg\exists m$ *s.t.* $\langle m, n\rangle \in E$; *i.e.*, attribute nodes are *leaves*. Moreover, according to the meaning of attribute nodes, each attribute node is *an only child:* $\forall n \in N_A$, $\forall e = \langle m, n\rangle \in E_A$, if $\exists e_1 = \langle m, n_1\rangle \in E$ then $n = n_1$.
- $\forall n \in N_D$ such that $\neg\exists e = \langle n, m\rangle \in E_R$ then $\exists e_1 = \langle n, m_1\rangle \in E_A$; *i.e.*, dimension nodes without concept children *must have an attribute child*. Indeed, black nodes must necessarily feature either some white children or the parameter node whose values are those of the corresponding subdimension. Thus, leaf nodes can only be either attribute nodes or white nodes.

A *chunk configuration* on the context dimension tree is expressed as a set of values (either white nodes or values of black nodes' parameters), one for each (sub)dimension; the values may be at any level in the tree. Note that sibling white nodes are mutually exclusive since they represent orthogonal concepts, while sibling black nodes represent the different (sub)dimensions which define a concept. Therefore, in descending the tree to build a chunk configuration, at each level, only one white node and any number of black siblings may be included.

Equation (1) shows one chunk configuration for tailoring data for a `visitor`. The data must be `human`-readable and are related only to the `site` the visitor is currently seeing. Moreover, the considered situation is a `routine` one. In this example, the `interest_topic` dimension has been instantiated with two white nodes, `non-expert` and `inscription` (values for *detail-level* and *typology*, respectively), that refine the `art-pieces` concept:

(1) ⟨{`visitor`}, {`non-expert`, `inscription`}, {`routine`}, {`human`}, {`this-site(VAR)`}⟩.

Note that some dimensions can be excluded: a chunk configuration with some defective dimension values means that the dimensions are not taken into account to tailor data, *i.e.*, the chunk corresponding to that configuration does not filter the data for those dimensions. For example, the meaning of not mentioning the value of the *time* dimension in Equation (1) is to keep the data related to any time interval.

The designer can also express constraints or preferences on the possible combinations of the dimension values; in fact, not all of them make sense for a given scenario; thus, if we combinatorially generate the complete set, many chunk configurations must be discarded. Hence, the tree model is enriched by introducing constraints in two forms:

1 *Forbid constraints*, described by the binary predicate *forbid*, allow the context designer to specify chunk configurations that are not significant, thus discarding those that would represent semantically meaningless context situations or be irrelevant for the application. For example, such a constraint can be established to forbid the `visitor` role's context from containing interest topics related to the `employees`' administrative data.

2 *Granularity-level constraints*, described by the binary predicate *granularity*, specify the level of detail of a dimension once the value of another dimension has been chosen. For example, the `supervisor` role is interested in a complete view of `art-pieces`; thus, we establish a granularity-level constraint between `supervisor` and `art-pieces`.

For each pair of nodes $n$, $m$ such that *forbid*($n$, $m$) (or *granularity*($n$, $m$)) is true, the following holds:

- $n \in N_C$ and $m \in (N_C \cup \overline{N_D})$, with $\overline{N_D} = \{n \in N_D \mid \neg\exists e = \langle n, m\rangle \in E_R\}$; the predicates related to constraints have as a first argument a concept node and, as a second argument, either a concept node or a dimension node without concept children
- $Descendant(n) \cap Descendant(m) = \emptyset$;[2] *i.e.*, constraints can connect only the nodes that are not a descendant of the other
- $\forall\langle n, m_1\rangle, \langle n, m_2\rangle$ with $m_1 \neq m_2$, such that:

  $$forbid(n, m_1) \wedge forbid(n, m_2)$$

  or

  $$granularity(n, m_1) \wedge granularity(n, m_2),$$

  then $Descendant(m_1) \cap Descendant(m_2) = \emptyset$; *i.e.*, two different constraints having as a first argument that the same node $n$ can only reach nodes which have no common subtrees
- $\forall\langle n_1, m_1\rangle$ such that *forbid*($n_1$, $m_1$), then $\neg\exists\langle n_2, m_2\rangle$ such that *granularity*($n_2$, $m_2$) with $n_2 \in Descendant(n_1) \cup \{n_1\}$ and $m_2 \in Descendant(m_1) \cup \{m_1\}$; *i.e.*, a granularity-level constraint cannot have as arguments two descendants of two nodes that are arguments of a forbid constraint
- $\forall\langle n_1, m_1\rangle$ such that *forbid*($n_1$, $m_1$), then it also holds that *forbid*($m_1$, $n_1$); *i.e.*, forbid constraints are symmetric.

Different formalisms can be adopted for specifying the context dimension tree, depending on the application environment it will be used in. For example, a DTD representation can be adopted when data are expressed in Extensible Markup Language (XML), while within an ontology-based global schema, the choice of an OWL representation (Curino *et al.*, 2006) might ease the association of the context dimension tree with the domain ontology.

The context dimension tree and the set of constraints are used to automatically compute all significant chunk configurations corresponding to the data views we want to generate. The tailoring process is guided by the tree structure: concepts of the information schema are associated with tree nodes expressing context facets. As a consequence, in the tailoring process, data are included in the chunk – *i.e.*, the (part of a) user view – if they are associated with the nodes that compose the current context.

Due to the non-mutually-exclusive nature of the black nodes, the number of chunk configurations grows more rapidly with the width of the dimension subtrees than with their depth. More precisely, a high number of black siblings (>3) causes the number of available values for a dimension to explode. Moreover, consider that, given a set of $n$ black sibling nodes, there are $2^n - 1$ subsets of those nodes,[3] each representing not only a valid dimension value, but also a valid *part* of a dimension value since each node may be further exploited by selecting one of its children. In the most frequent case, at least one node is specified per dimension, so the worst-case number of chunk configurations (not considering those discarded by constraints) is given *by the product of the number of possible values per dimension*. As a consequence, the generation of all chunk

configurations is not a too complex task and has been automated; the attention is rather on the number of produced chunk configurations, which are the inputs of the subsequent (largely manual) steps of the data view design methodology. However, it is worth noting that the *interest_topic* dimension is often very detailed, with a rich subtree generating itself a high number of possible values while the other dimensions are less deep. Our test cases have been characterised by context dimension trees similar to the one presented in Figure 3 which, after considering the constraints, led to a number of meaningful chunk configurations in the order of 300. In our experience, this is an affordable number of elements to work with.

## 5 View definition and knowledge tailoring

Here, we briefly exemplify how the context model of the running example can be employed to design views over the archaeological ER schema of Figure 1.

Once the chunk configurations have been (possibly combinatorially) generated, the next task is the most intriguing one, since it consists of associating each chunk configuration with the definition of the corresponding schema chunk. Such a (set of) chunk(s) forms the personalised view for the given information user in the context identified by a (set of) chunk configuration(s).

Such associations, along with the methodological steps to produce the resulting view, are detailed in Bolchini *et al.* (2007) for the ambient array context model. However, for the moment, the association is hand-made by designing, for each chunk configuration, the view (query) defining the corresponding chunk. Each chunk configuration is associated by the designer to the data portion relevant to the corresponding context and then the system automatically generates the query which will tailor the corresponding data from the actual dataset.

Owing to space constraints, we present only the result of the association of relevant areas (of the source schema) with respect to the following chunk configuration:

⟨{`operator(VAR)`}, {`specialist`, `foundation`}, {`routine`}, {`this-site`}, {`today`}⟩.

The corresponding view on the global schema is the area bordered with a dotted line in Figure 1. Indeed, the chunk configuration identifies the portion of data relevant for the specific operator `operator (VAR)` (*VAR* being the identifier), who has to work `today` and in `this-site` on a certain number of artworks. The operator monitors the state of the art pieces and fills or updates entries to be made available to the scientific institutions and, at a less specialised level, to the general public. For such art pieces, the chunk configuration further selects those that refer to foundations (`foundation`) and, as far as documentation is concerned, selects expert-level data (`specialist`). This operation is performed by the designer for each significant chunk configuration, thus leading to the definition the context-aware data views for all the possible contexts. It is worth noting that the context dimension tree allows a more precise tailoring of the schema with respect to the flat array model due to the possibility of filtering at a finer granularity.

## 6 Conclusions and future work

Our context model has been defined to support the knowledge tailoring task; thus, the following aspects have been considered as important drivers:

- the context dimension tree is formally defined as a tree that is fully independent of the specific representation paradigm, which may be XML (DTD or XML schema) or ontologies (OWL).
- By introducing appropriate constraints between different aspects of a context description, the model allows representing all the useful contexts for an application scenario and discard the meaningless ones.
- The context dimension tree represents contexts with different levels of abstraction. Indeed, the set of chunk configurations can be formalised as a partially ordered algebraic structure where the greatest (top) chunk configuration corresponds to the entire source schema (*i.e.*, 'no value per dimension'). These levels of abstraction have an important impact on the tailoring task that can be performed by considering different policies in selecting sets of relevant data on the basis of the different levels of detail.

As a conclusion, we have introduced in our context model the features and possibilities that were not offered or not powerful enough in other models, but are important to support the data tailoring task in all its phases.

We are currently working on an extension of the methodology where the designer associates nodes of the context dimension tree to areas of the global schema and the system derives automatically the complex views associated with chunk configurations. However, this process must be supervised since the application semantics might require sophisticated associations.

More advanced features such as adequate treatment of the context history, reasoning and learning features and ambiguity management, though dispensable, will be investigated for further developments.

## References

Aberer, K., *et al.* (2004) 'Emergent semantics: principles and issues', Invited paper at the *9th Int. Conf. on Database Systems for Advanced Applications (DASFAA 2004)*, Jeju Island, Korea, LNCS 2973, Springer-Verlag, March, pp.25–38.

Agrawal, R. and Wimmers, E.L. (2000) 'A framework for expressing and combining preferences', *Proc. ACMSIGMOD Int. Conf. on Management of Data 2000*, ACM, pp.297–306.

Baldauf, M., Dustdar, S. and Rosenberg, F. (2004) 'A survey on context-aware systems', *Int. Journal of Ad Hoc and Ubiquitous Computing*.

Bolchini, C., Curino, C., Quintarelli, E., Schreiber, F.A. and Tanca, L. (2006a) 'Context-addict', Technical Report 2006.044, Dip. Elettronica e Informazione, Politecnico di Milano.

Bolchini, C., Curino, C., Schreiber, F.A. and Tanca, L. (2006b) 'Context integration for mobile data tailoring', *Proc. 7th IEEE/ACM Int. Conf. on Mobile Data Management*, Nara, Japan, 9–13 May, p.5.

Bolchini, C., Schreiber, F.A. and Tanca, L. (2004) 'A context-aware methodology for very small data base design', *SIGMODRecord*, Vol. 33, No. 1, pp.71–76.

Bolchini, C., Schreiber, F.A. and Tanca, L. (2006c) 'Data management', Chap. 6 in B. Pernici (Ed.) *Mobile Information Systems – Infrastructure and Design for Adaptivity and Flexibility*, Springer, pp.155–176.

Bolchini, C., Schreiber, F.A. and Tanca, L. (2007) 'A methodology for very small database design', *Information Systems*, March, Vol. 32, No. 1.

Buchholz, S., Hamann, T. and Hübsch, G. (2004) 'Comprehensive Structured Context Profiles (CSCP): design and experiences', *Proc. 2nd IEEE Conf. on Pervasive Computing and Communications Workshops*, Orlando, Florida, USA, 14–17 March, pp.43–47.

Chen, G. and Kotz, D. (2000) 'A survey of context-aware mobile computing research', Technical report, Department of Computer Science, Dartmouth College, Hanover, New Hampshire, USA.

Chen, H., Finin, T. and Joshi, A. (2003) 'An intelligent broker for context-aware systems', *Adjunct Proc. of Ubicomp 2003*, October, pp.183–184.

Chen, H., Perich, F., Finin, T. and Joshi, A. (2004) 'SOUPA: Standard Ontology for Ubiquitous and Pervasive Applications', *Int. Conf. on Mobile and Ubiquitous Systems: Networking and Services*, August.

CoDaMoS development team (2003) *The CoDaMos Project*.

Curino, C., Quintarelli, E. and Tanca, L. (2006) 'Ontology-based information tailoring', *Proc. 2nd IEEE Int. Workshop on Database Interoperability (InterDB 2006)*, April, p.5.

De Virgilio, R. and Torlone, R. (2005) 'A general methodology for context-aware data access', *Proc. ACM Int. Workshop on Data Engineering for Wireless and Mobile Access*, Baltimore, Maryland, USA, 12 June, pp.9–15.

De Virgilio, R., Torlone, R. and Houben, G-J. (2006) 'A rule-based approach to content delivery adaptation in web information systems', *Proc. 7th IEEE/ACM Int. Conf. on Mobile Data Management*, Nara, Japan, 9–13 May, p.21.

Fahy, P. and Clarke, S. (2004) 'CASS – middleware for mobile context-aware applications', *Proc. Mobisys 2004 Workshop on Context Awareness*, Boston, Massachusetts, USA, 6 June.

Gu, T., Keng Pung, H. and Zhang, D.Q. (2005) 'A service-oriented middleware for building context-aware services', *Journal of Network and Computer Applications*, Vol. 28, No. 1, pp.1–18.

Jiang, L., Topaloglou, T., Borgida, A. and Mylopoulos, J. (2006) 'Incorporating goal analysis in database design: a case study from biological data management', *Proc. 14th Italian Symp. on Advanced Database Systems*, Portonovo (Ancona), 18–21 June, pp.14–19.

Kaenampornpan, M. and O'Neill, E. (2004) 'An integrated context model: bringing activity to context', *Proc. Workshop on Advanced Context Modelling, Reasoning and Management*, Nottingham, England, 7–10 September.

MAIS Project (2002–2005) 'MAIS: multi channel adaptive information system'.

Ouksel, A.M. (2003) 'In-context peer-to-peer information filtering on the web', *SIGMOD Record*, Vol. 32, No. 3, pp.65–70.

Petrelli, D., Not, E., Strapparava, C., Stock, O. and Zancanaro, M. (2000) 'Modeling context is like taking pictures', *Proc. of the Workshop "The What, Who, Where, When, Why and How of Context-Awareness" in CHI2000*, The Hague, The Netherlands, 1–6 April.

Preuveneers, D., *et al.* (2004) 'Towards an extensible context ontology for ambient intelligence', *Proc. 2nd European Symp. Ambient Intelligence*, Lecture Notes in Computer Science, Springer, pp.148–159.

Rajugan, R., Chang, E. and Dillon, T.S. (2006) 'Ontology views: a theoretical perspective', *On the Move to Meaningful Internet Systems 2006: OTM 2006 Workshops*, Lecture Notes in Computer Science, Springer, Vol. 4278, pp.1814–1824.

Sridharan, H., Sundaram, H. and Rikakis, T. (2003) 'Computational models for experiences in the arts, and multimedia', *Proc. ACM Workshop on Experiential Telepresence*, Berkeley, California, pp.31–44.

Strang, T. and Linnhoff-Popien, C. (2004) 'A context modeling survey', *1st Int. Workshop on Advanced Context Modelling, Reasoning and Management*, Nottingham, England, 7–10 September.

Strimpakou, M., Roussaki, I. and Anagnostou, M.E. (2006) 'A context ontology for pervasive service provision', *20th Int. Conf. on Advanced Information Networking and Applications (AINA 2006)*, Vienna, Austria, 18–20 April, pp.775–779.

Torlone, R. and Ciaccia, P. (2003) 'Management of user preferences in data intensive applications', *Proc. of the 11th Italian Symp. on Advanced Database Systems*, SEBD, pp.257–268.

Wouters, C., Rajugan, R., Dillon, T.S. and Rahayu, J.W.R. (2005) 'Ontology extraction using views for semantic web', in *Web Semantics and Ontology*, Idea Group Publishing, pp.1–40.

Yang, S.J.H., Huang, A.F.M., Chen, R., Tseng, S-S. and Shen, Y-S. (2006) 'Context model and context acquisition for ubiquitous content access in ULearning environments', *IEEE Int. Conf. Sensor Networks, Ubiquitous, and Trustworthy Computing*, Vol. 2, pp.78–83.

## Notes

1 See Petrelli *et al.* (2000) for an example.

2 We use the *Ancestor* or *Descendant* functions to compute the set of ancestors or descendants of a node with respect to *E*.

3 The empty subset is not significant in this situation.