Chapter 12
Context-Aware Knowledge Querying in a Networked Enterprise

Cristiana Bolchini, Elisa Quintarelli,
Fabio A. Schreiber, and Maria Teresa Baldassarre

Abstract. In today's knowledge-driven society, the increasing amount of heterogeneous information, available through a variety of information channels, has made it difficult for users to find the right information at the right time and at the right level of detail. This chapter presents the exploitation of context-awareness in order to reduce the plethora of information that,within the networked enterprise scenario, may confuse and overwhelm the users in need of high-quality information coming from the several available and heterogeneous information sources.

12.1 Introduction

A networked enterprise is a scenario rich of information and data, to be shared among different users, each one characterised by a specific role and interests. To solve the information overload problem, researchers have developed systems that adapt their behaviour to the goals, tasks, interests, and other characteristics of their users. The notion of context, emerged in the past years in various fields of research, has more recently received a lot of attention also in the computer science field, because it can help distinguishing useful information from noise, i.e., from all the information not relevant to the specific application; indeed, the same piece of information can be considered differently, even by the same user, in different situations, or places – in a single word, in a different context.

In this chapter we present the methodology and system architecture adopted in ART-DECO to support all the phases of contextual view design: (a) the specification

Cristiana Bolchini · Elisa Quintarelli · Fabio A. Schreiber
Politecnico di Milano, Dipartimento di Elettronica e Informazione, P.zza L. da Vinci, 32, Milano - I20133 Italy
e-mail: {bolchini,quintare,schreiber}@elet.polimi.it

Maria Teresa Baldassare
Dipartimento di Informatica - Università degli Studi di Bari "'Aldo Moro"'
e-mail: baldassarre@di.uniba.it

of a context model appropriate for the specific application; (b) the association of each previously defined context with the portion of data relevant to that specific context; (c) the access to contextualised data. We show its applicability in the wine production scenario, where several classes of users access a variety of information, made available from the networked enterprise data sources, from sensors used for monitoring the productive cycle and from external sources. The access to such information is offered through a Context-Aware Web Portal, where the users identify themselves by logging into the system and defining their context by answering a short sequence of pre-defined questions. This initial identification phase aims at semi-automatically determining the active context of the user, in order to customise the portion of information made available for browsing and querying. These context-related behaviours (context specification and context-aware tailoring of data) are the outputs of a design phase where, taking into account the application scenario, the possible contexts are modeled and their relationship with the available data is designed. We focus the attention on the elements of context-based access to data, presenting the various facets of the approach. In particular, in Section 12.2 we present the context related aspects of the architecture, followed in Section 12.3 by the description of the model we use for defining the possible application contexts. Sections 12.4 and 12.5 are devoted to how the context-aware subsystem supports data tailoring and access to the various kinds of data sources, respectively. Section 12.6 shows an application of the theoretical concepts, illustrated in the previous sections, to a real context consisting in a wine production scenario through a Context-Aware Web Portal.

12.2 System Architecture

This section presents the context-aware subsystem of the overall system architecture (shown in Figure 12.1), focusing the attention on the **Context-Aware Web Portal**, the **Design-Time Context Manager** subsystem, the **Context-Aware Application Server**, enabling the context-aware technology, to offer the users the possibility to filter the data (available in the **Data Access Server**) s/he will access. The architecture is the one presented in Chapter 21, analyzed from the point of view of the context-aware data management and access.

The upper layer of this facet of the architecture consists of the Context-Aware Web Portal, which allows the user to access the information requested based on her/his role and on the context. The portal represents the access point for potential users to the data, in that it provides a specific view on a subset of data, based on the role each user has within the particular context, avoiding the need to dodge through not relevant information. Furthermore, all of the elaborations, queries and data retrieval operations necessary to present the information in the web portal are transparent to the user. Indeed, the communication between portal and the system occurs through the web which filters the requests and breaks them down with respect to the single architectural components involved. This occurs independently of the type of information source, either internal, external or coming directly from sensors.

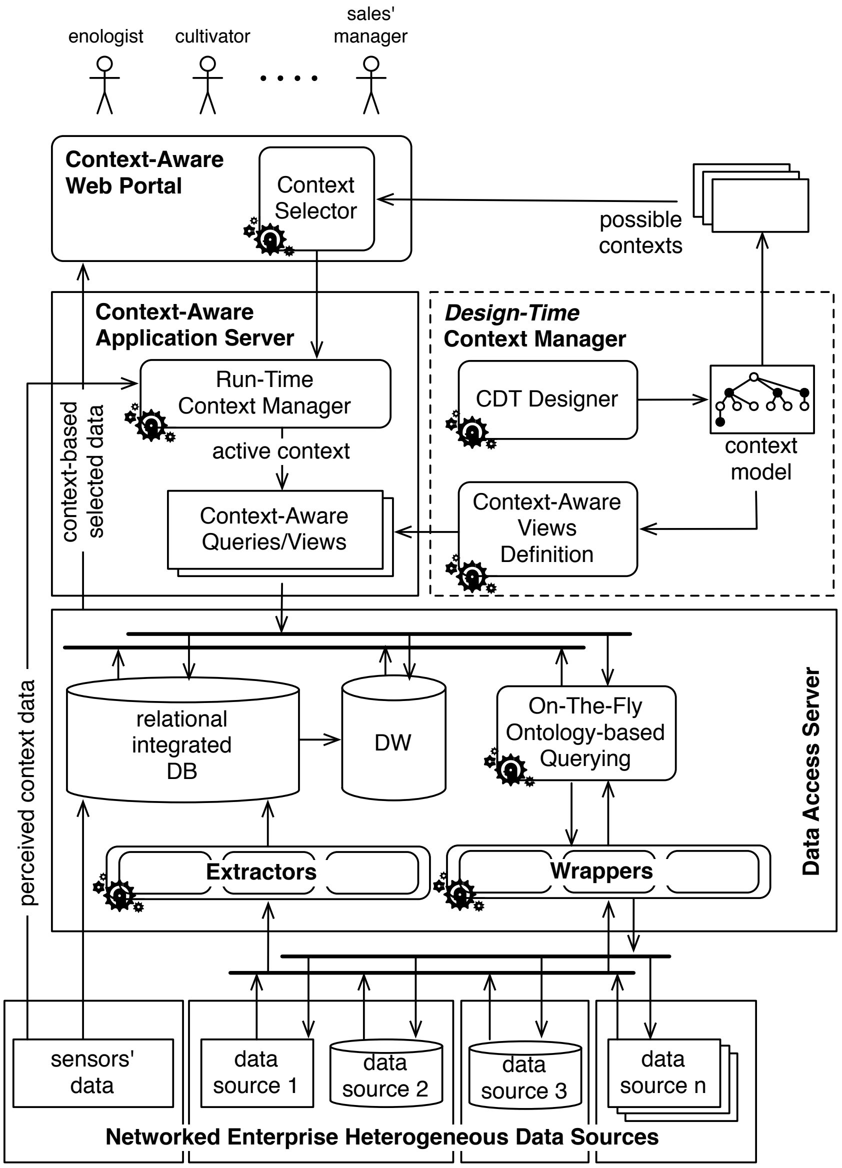


Fig. 12.1 The system architecture of the networked enterprise, from the context-aware data access point of view

The rest of the architecture consists of the above mentioned components, devoted to the following tasks:

Design-Time Context Manager: In charge of supporting, at design time, the modelling of the context of the specific application scenario, and the formulation of the context-aware views and queries over the available data sources, associated with the various possible contexts.

Context-Aware Application Server: In charge of determining the *active context* based on the explicit user's selection and on the contextual parameters autonomously perceived from the environment, dispatching the queries to the different data sources and delivering to the web portal and to the other applications the context-driven retrieved data.

Data Access Server: In charge of maintaining all available data and of the data access technologies.

Indeed, context is taken into account both at *design time* and at *run time*. In the former phase, the Design-Time Context Manager supports the designer in the specification of the possible contexts the users will be acting in. This task is performed by means of the so-called **CDT Designer**, a tool for the specification of the context model according to the adopted formalism, the *Context Dimension Tree*, CDT [2], briefly presented in Section 12.3. The second aim of the Design-Time Context Manager is the definition of context-aware views and queries over the data sources. More precisely, it is necessary to define, for each possible context, which is the context-relevant portion of data; the overall goal is to filter unnecessary information, that would make the exploitation of interesting data more difficult. Therefore, the **Context-Aware Views Definition** module automatically computes such views and queries, starting from the CDT and additional information, according to the methodology presented in [5, 6] for XML and relational data sources. More information on how context affects data querying is presented in Section 12.5, by referring to the different kinds of data sources the application scenario may be accessing. Furthermore, data to be accessed can reside in the enterprise relational database, as the result of a previously performed import by means of format-specific extractors; these components wrap the enterprise-interesting data sources to extract the (possibly external) information to be integrated and materialised within the enterprise relational database. Other external data sources, possibly not known in advance, are accessed on-the-fly and directly queried by means of wrappers, used to interpret the external data schemata, to translate the context-aware query into the native data source language and to retrieve the corresponding information. Should it be necessary to integrate the retrieved data, the Data Access Server offers such a feature to provide a unified answer to the user's data request. Altogether these modules constitute the Data Access layer and the approaches to implement them are further discussed in Section 12.5.

The information on the supported contexts, the context-aware views and queries is exploited at run time, by means of the **Context-Aware Application Server**; two are the main tasks this component carries out: a) it determines the active context, and, on its basis, b) it applies the context-aware views and queries, collecting the data from the sources and providing it to the upper application layer. In particular, the **Run-Time Context Manager** is in charge of receiving both user's explicit characterization (e.g, his/her role, specific interest category) and the perceived context data (for instance, from a GPS or other sensors) to determine the active user's context, used to select the interesting portion of data, by applying the appropriate **Context-Aware Queries/Views**.

The application layer, the **Context-Aware Web Portal** in our scenario, supports the selection of the user's active context, to allow a customised access to data. In particular, some of the aspects characterizing the active context can be automatically derived (such as the location and time of the user, possibly her/his role within the application scenario, the kind of device used to access the information), while others may require an explicit specification by the user (for instance, her/his interests). Therefore, the output of the Design-Time Context Manager component is the list of possible contexts, together with the different aspects characterising each of them. In particular, the context model is such that it is quite immediate to derive an application related **Context Selector**, taking as input the possible contexts and allowing the user to specify her/his active one.

In the next sections, we shall detail the several aspects contributing to the overall picture by starting from the context model, which is the pivot element of the entire approach.

12.3 Context Model

Our context model, called Context Dimension Tree, plays a fundamental role in tailoring the available information according to the user's needs. In particular, the CDT models all possible applicable contexts with respect to the given application scenario, and is then used as the starting point to define which portion of data should be made available in each possible context. It is worth noting that in different working scenarios it is possible to associate with a context not just the relevant portion of data, but, for instance, a set of rules or actions that should be carried out. In this perspective, the proposed context model is a quite general means for context-aware knowledge management.

Figure 12.2 shows an example – discussed below – which models the possible contexts of the wine cultivation scenario used in in Chapter 21. Black nodes represent dimensions and sub-dimensions, white nodes represent dimension values; nodes with a double border are parameters.

In the CDT, the root's children are the context main dimensions, which capture the different characteristics of the users and of their context; by specifying a value for each dimension, a point in the multidimensional space representing the possible

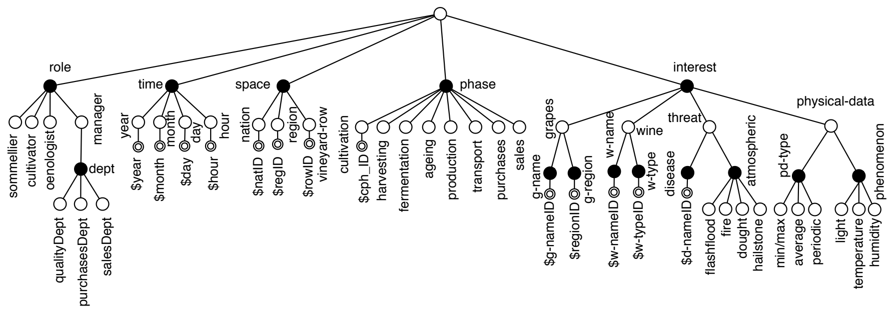


Fig. 12.2 A context model – CDT– for the wine case study

contexts is identified. As an alternative, rather than explicitly listing all possible values, in order to specify the exact value for the (sub-)dimension, it is possible to specify a parameter to be given an actual value at run time. Moreover, a dimension value can be further analysed and specialised, becoming the root of another sub-tree.

Independently of the application scenario, one of the key dimensions is the `role`, for the specification of the possible classes of users of the application; it is quite common, in fact, that different classes of users are interested in different portions of data, possibly due to access permission limitations. In the present example, seven are the foreseen roles of users accessing the available data, namely the wine *cultivator*, the *oenologist*, the *sommelier* and the *managers*, further classified on the basis of their specific department. Different, possibly partially overlapped, subsets of data will be made available to users of the various classes; when the user identifies him/herself as belonging to one of these context classes, a dedicated tailoring of the information will be provided.

`time` and `space` dimensions are often included in the context model, used to tailor data according to the time the user accesses the information or the time-window data belongs to, or to the user's location. In the wine scenario, as far as the temporal aspects are concerned, different granularities are introduced, based on the fact that certain information is analysed on a per-*year* or per-*month* scale, as for instance sales or production in general, whereas, when accessing data related to the productive process, a more frequent access is necessary (*daily* or *hourly*). Similarly, also the location can be used to filter information on wine, for instance with reference to a specific production *nation* or *region*.

The `phase` dimension is used to model the different application situations the user may be in; the values for this dimension include standard commercial phases such as *purchases* (e.g., of raw components such as sugars and yeast) and *sales* (related to the amount of distributed and sold wine), along with domain-specific phases like *cultivation* (further identified by its specific $*cph_ID*) and *harvesting* of grapes, *fermentation* of grape must, and *ageing*, and *transportation* of wine. While the commercial phases are, in general, used to tailor data stored in the commercial database, the latter dimensions are mostly used for the analysis of the plants growing process and the environmental phenomena that may be collected from sensors. Notice that the selection of a particular cultivation phase is important since certain values perceived by sensors may be considered as “alarms” or not on the basis of the cultivation phase of the grapevine.

Finally, the `interest` dimension constitutes the other key dimension, being the one that mainly drives the tailoring process, since it coarsely partitions data into chunks, further refined by the other dimension values. Given the coarse granularity, a hierarchy of levels is introduced, to refine the selection criteria, for a more precise tailoring of data. In the present example, four main interest areas have been envisioned, namely *grapes*, *wine*, *threat* and *data-from-sensors*. In order to allow a more refined selection of information, *grapes* can be selected according to their name (*g-name*) and region (*g-region*), *wine* by name (*w-name*) and (*w-type*). As far as *physical-data* is concerned, it is selected based on the physical *phenomenon* they monitor and on the kind of monitoring (*pd-type*) the user is interested in. Moreover, the *threat* value is selected whenever the analyst is concerned with those phenomena that might threaten the quality of the final product. Threats include plant *disease* and strong *atmospheric* phenomena like *flashflood*s, prolonged *drought*, *hailstone* and *fire*, provoked either by natural causes (e.g., hot temperatures) or arsons.

A context derives from the specification of one or more of its characterizing dimensions, whereas given a dimension, only one of its values can be selected to define a single context. For example, given the CDT of Figure 12.2, the following is a valid context:

$$\begin{aligned} C \equiv\ & (\mathtt{role} = \mathtt{oenologist}) \\ \wedge\ & (\mathtt{time} = \mathtt{year}(\$\mathtt{yearID} = 2010)) \\ \wedge\ & (\mathtt{space} = \mathtt{region}(\$\mathtt{regID} = \text{“}\mathtt{Sicily}\text{”})) \\ \wedge\ & (\mathtt{phase} = \mathtt{production}) \\ \wedge\ & (\mathtt{interest} = \mathtt{grapes}(\$\mathtt{regionID} = \text{“}\mathtt{Sicily}\text{”})) \end{aligned}$$

Quite intuitively, the context refers to an oenologist interested in production data of grapes in year 2010 and grown in Sicily. White nodes and black leaf nodes are also called *context elements*, since they constitute the the characterizing elements of a context.

The model allows the expression of *constraints* or *preferences* among the values of a context definition to avoid the generation of meaningless ones (more details on constraints and their use can be found in [6]). As an example, a constraint might indicate that, in a context for a user with a sale manager role (*dept=salesDept*), information referring to data located in a certain vineyard row (*vineyard-row*) is not interesting. This is stated as follows (*useless-context* constraint, `uc`):

$$\begin{aligned} & \mathtt{uc}((\mathtt{dept} = \mathtt{salesDept}), (\mathtt{space} = \mathtt{vineyard\text{–}row})) \equiv \\ & \quad \neg((\mathtt{dept} = \mathtt{salesDept}) \wedge (\mathtt{space} = \mathtt{vineyard\text{–}row})) \end{aligned}$$

Here we have introduced a set of constraints to separate the two main areas of the application domain, which are related to the wine production process, and to its output as distributed and sold.

A *context*, stemming from this model, is obtained by composing elements of the CDT, and determines a portion of the entire data set, i.e., a *view*, specified at design-time. At run-time, when a context becomes active, the corresponding view is selected and all parameter values involved in the specification of that context are appropriately instantiated.

12.4 Methodology for Context-Aware Data Filtering

The ultimate goal of context-awareness in the ART-DECO project is to filter knowledge and information based on the user's active context. More precisely, given the possible and reasonable contexts modelled by the CDT, it is then necessary to associate with each of them the portion of data to be made available to the user. We have proposed a methodology for semi-automatic computation of the portion of data associated with each possible context, starting from the context model and the schema of the application data. Such a methodology consists of an approach of general validity, specialised for the different nature of the available data sources. The data and application designer is in charge of defining such a relation, at design time, so that, at run-time, once the user's active context is known, s/he receives the selected portion of information.

Two strategies have been proposed to support the designer in his/her activity.

The first strategy works globally on a *per-context* basis, it is precise but burdensome, and it is quite independent of the nature of the data source, since it requires the designer to specify, for each possible significant context, the portion of the data to be considered relevant. In such a perspective, the definition of the so-called *relevant area* is carried out independently for each context, by specifying on the data source what elements are to be discarded and filtered. The activity is though quite demanding, especially because the number of possible contexts derived even from small CDTs is of the order of hundreds. As a result, although many contexts might have similar associated relevant areas, the required designer's effort is particularly high. On the other hand, a great advantage of such an approach is the possibility to precisely tailor the interesting data, with a fine granularity.

The second strategy, working on a *per-context-element* basis (the node of the CDT), has been introduced to cope with the complexity of the former one, to reduce the amount of user intervention, still achieving a relevant filtering power and precision. This approach requires the designer to specify, for each dimension value (white nodes) or dimension node with parameter (black nodes with parameter), the relevant area that should be made available when a context contains such an element. Subsequently, the relevant area corresponding to an entire context is automatically computed by combining the areas of the included context elements, by means of ad-hoc designed operators (see [6] for the relational scenario).

For example, the contextual view (relevant area) for the context *C* introduced above is obtained by combining the relevant areas of its components, that are

(`role` = `oenologist`), (`time` = `year`(`$yearID` =`2010`)),
(`space` = `region`(`$regID` =“`Sicily`”)), (`interest` = `grapes`).

In this perspective, the methodology is strictly related to the specific nature of the data source, since operators for combining *partial* relevant areas cannot be general-purpose because they operate on the source schema and instances. Indeed, when dealing with relational data sources, the partial relevant areas are expressed by means of sets of relational views, thus, the intersection operator (a set-intersection operator, dubbed *double intersection*) is applied to sets of virtual tables (views) both at the schema and the instance levels, between pairs of relational views defined on the same relation, thus producing a tailored portion of the database containing only the pertinent common information.

In this chapter we focus on the relational database scenario, used to tailor the enterprise internal database. For dynamically querying external data sources an ontological approach is proposed (see Chapter 11).

12.5 Data Querying

In the ART-DECO scenario we face a situation characterised by three main categories of information sources: i) *internal*, resulting from the extraction of data from different heterogeneous sources, integrated and materialised in a unified relational database, ii) *external*, unknown in-advance and possibly transient data sources, to be queried on the fly, without materializing their information in a permanent data storage, and iii) data coming *from sensors*, used to perceive part of the user's context.

The internal data source is the result of an extraction and integration process carried out on the information coming from the available and stable data sources of the networked enterprise. A set of *extractor*s has been designed (Figure 12.1), to bundle the different sources according to their native format, and to extract their information, to be saved in a general, unique relational database. The proposed context-aware data filtering approach accesses this relational data, by selecting a portion of the entire information, according to the user's context. The filtering process, discussed in Section 12.5.1, adopts the per-context-element strategy discussed in Section 12.4, and exploits the use of specific operators to combine the relevant areas of information to determine the portion of the entire database corresponding to each possible context.

When data sources are not stable and possibly not available at all times, a more dynamic access to their data is proposed. In this situation, a set of *wrappers* has been designed and developed, to cope with the different source formats; these wrappers offer the opportunity to interpret the source data, to deliver the query to such sources, and to collect the retrieved information. Section 12.5.2 offers an overview of the adopted approach.

Finally, context awareness has a two-fold role with respect to sensors' data, as discussed in Section 12.5.3: part of the collected information is stored in the integrated internal database, to constitute part of the information to be made available through the application and is affected by context as in the general internal case, part of it is directly used to determine some of the contextual parameters, such as the time of the day or the location (e.g., from GPSes), that is it becomes part of the context itself. In the latter situation, the information is directly exploited by the *Run-Time Context Manager* to actualise the correct context.

12.5.1 Internal Data Sources

Data extracted from the several available sources are integrated and materialised into a unique relational database for a context-based filtered access, as discussed in the following. Moreover, for an effective and efficient analysis of trends, a data warehouse has been developed. Even though the nature of the data is common to both situations, context-awareness can be exploited with different flavours.

Once the context model for the specific application scenario has been defined, the designer must then define the association between contexts and their relevant portion of data. Here we only discuss the strategy requiring the definition of a relevant area for each one of the context elements, to be combined to derive the relevant area for the whole context. In particular, two steps constitute the adopted methodology: i) *partial view* definition and ii) *partial view* combination.

During the former activity, the designer defines a mapping between each *context element* of the CDT and a portion of the relational schema, a *partial view* or *relevant area*. More precisely, the partial view associated with a context element has a tailored schema with respect to the global one, where some relations do not appear and those that appear may have a reduced number of attributes or tuples. In fact, in this operation the designer is defining which subset of the information is of interest any time a context includes such a context element.

As an example, the designer must specify the relevant area, $\mathcal{R}el(\texttt{oenologist})$, associated with the context element *oenologist*, by determining which relations contain data interesting for an oenologist, and with respect to such relations, which attributes, and, eventually, which subset of tuples. The same activity must be carried out for each one of the white nodes in the tree and all other black nodes with a parameter, since they can be part of a context specification.

Once all the partial views have been specified, the second step automatically combines the partial views of the nodes constituting a given context, to obtain a *final view*, which is a tailored schema and dataset over the original one. The computation of the combined view is carried out by means of purpose-defined operators, introduced preliminarily in [6], working on sets of relations. For example, the contextual view (relevant area) for the context C introduced above, is obtained by combining by means of the so-called *double-intersection* operator, $\Cap$, the partial relevant areas of its components, that is:

$$\begin{aligned}\mathcal{R}el(C) = {} & \mathcal{R}el(\texttt{role} = \texttt{oenologist}) \Cap \mathcal{R}el(\texttt{time} = \texttt{year}(\$\texttt{yearID} = 2010)) \Cap \\ & \Cap \mathcal{R}el(\texttt{space} = \texttt{region}(\$\texttt{regID} = \text{“}\texttt{Sicily}\text{”})) \Cap \\ & \Cap \mathcal{R}el(\texttt{phase} = \texttt{production}) \Cap \mathcal{R}el(\texttt{interest} = \texttt{grapes})\end{aligned}$$

Consider now the following partial view defined by the designer, associated with the grapes *interest*, containing information about vineyards, their composition w.r.t. grapevines, and the harvest data:

$$\mathcal{R}el(\texttt{interest} = \texttt{grapes}) = \{\textsc{Vineyard}, \textsc{Harvest}, \textsc{Cellar}, \textsc{Row}, \textsc{Grapevine}\}$$

The partial view, associated with the region($regID =“Sicily”) *space*, includes information about wines, vineyards, and so on, filtered on the base of the specific region:

$$\begin{aligned}\mathcal{R}el(\texttt{space} = \texttt{region}(\$\texttt{regID} = \text{“}\texttt{Sicily}\text{”})) = \{ & \sigma_{region=\text{“}Sicily\text{”}}\textsc{Vineyard}, \\ & \textsc{Harvest} \ltimes_{vineyard\_id=v\_id} \sigma_{region=\text{“}Sicily\text{”}}\textsc{Vineyard}, \\ & \textsc{Cellar} \ltimes_{vineyard\_id=v\_id} \sigma_{region=\text{“}Sicily\text{”}}\textsc{Vineyard}, \\ & \textsc{Barrel} \ltimes_{cellar\_id=cl\_id} \textsc{Cellar} \ltimes_{vineyard\_id=v\_id} \sigma_{region=\text{“}Sicily\text{”}}\textsc{Vineyard}, \ldots\}\end{aligned}$$

The $\Cap$ operator applied on partial views gives as result a set of tables obtained by applying the classical intersection operator $\cap$ to all pairs of expressions having the same schemata. On the above partial views we obtain the following set:

$$\begin{aligned}& \mathcal{R}el(\texttt{interest} = \texttt{grapes}) \Cap \mathcal{R}el(\texttt{space} = \texttt{region}(\$\texttt{regID} = \text{“}\texttt{Sicily}\text{”})) = \\ & \{\sigma_{region=\text{“}Sicily\text{”}}\textsc{Vineyard}, \textsc{Harvest} \ltimes_{vineyard\_id=v\_id} \sigma_{region=\text{“}Sicily\text{”}}\textsc{Vineyard}, \\ & \qquad \textsc{Cellar} \ltimes_{vineyard\_id=v\_id} \sigma_{region=\text{“}Sicily\text{”}}\textsc{Vineyard}\}\end{aligned}$$

As another example, the contextual view for a *sommelier*, interested in the *average temperature* of the canteen for the current day, while the wine is *ageing* is computed as the intersection of the relevant areas of the context elements involved in the definition, that is:

$$\begin{aligned}\mathcal{R}el(C) = {} & \mathcal{R}el(\texttt{role} = \texttt{sommelier}) \Cap \mathcal{R}el(\texttt{time} = \texttt{day}(\$\texttt{dayID} = \texttt{today}())) \Cap \\ & \Cap \mathcal{R}el(\texttt{phase} = \texttt{ageing}) \Cap \mathcal{R}el(\texttt{phenomenon} = \texttt{temperature}) \\ & \Cap \mathcal{R}el(\texttt{sd–type} = \texttt{temperature})\end{aligned}$$

It is immediate to see that the number of context elements for which the designer has to define a relevant area is at least an order of magnitude smaller than the number of possible contexts, therefore the per-context-element strategy is particularly efficient. In general, the designer can start by the top context elements in the tree (the ones closer to the root) and define the relevant area for each one of them. Nodes belonging to sub-trees rooted in such upper nodes will have a relevant area that is included in their parent node's area, thus limiting the overall necessary effort. As previously stated, since the combining procedure is automatic starting from the definition of the relevant areas, the resulting contextual view might be slightly imprecise, however the designer can finally refine it to satisfy his/her needs.

12.5.2 External Data Sources

In the ART-DECO scenario, Semantic Web technologies such as RDF [8] and OWL [17], which might fail on large-scale data integration [11], play an important role as tools for on-the-fly integration of small pieces of information belonging to external, independent and heterogeneous data sources. In open environments, *external data sources* may be Relational DBMSs, XML and RDFS files, web-pages, natural language documents, sensor data, etc. Sensor data are taken care of by the PerLa query language and by appropriate middleware layers (see Chapter 18), while webpages and natural language documents are extracted according to technologies described in Chapters 10 and 9. ART-DECO provides a framework to automatically extract ontological representations from the schemas of Relational databases, XML and RDFS files, and (semi)automatically map them to the application-domain ontology (see Chapter 9), which is used as a uniform representation of the available information and can be queried using SPARQL [13]. Moreover, context-based information space reduction is performed at runtime by selecting fragments of the application-domain ontology which, through the mappings, induce corresponding fragments on the data-source ontologies, and thus a subset of the information space. A complete description of our approach to on-the-fly query distribution, with a focus on relational datasources, is provided by Chapter 13, while here we give a quick overview.

We consider ontologies expressed using CA-$\mathcal{DL}$, a fragment of OWL-DL [17] which (a) can uniformly represent the data and the user context(s) in any application domain and (b) supports context-aware SPARQL query-answering and distribution to (possibly heterogeneous) information sources. The SPARQL query is translated into a context-aware query over the (context-aware) portion of the domain ontology, and thus reduced. Then, the distribution of the reduced query takes place, as described in Chapter 13. All the data-source queries are translated into the native language of each target source by means of automatically generated wrappers, which also provide a CA-$\mathcal{DL}$-compatible representation of the answers. A final step hands the results over to the user after a (possible) further contextual refinement.

12.5.3 Data from Sensors

Thanks to the advent of pervasive systems, sensors became an important source of data; sensors are nearly always organised in networks and often they are tiny devices which communicate over wireless links which constitute a *wireless sensors network* (WSN). From a datacentric point of view a WSN can be seen as a special form of *distributed database* [16]. However, differently from traditional (relational) distributed databases, in which data are permanently stored on stable storage and data sources are mostly known and durable, in a WSN *sources availability greatly vary over location and time* due to one of the most outstanding properties of wireless sensors: their life is bounded to that of their power source. Other reasons for sensors data unavailability lie in the weakness of the communication links and on the physical integrity of the sensor itself, which is often deployed in harsh environments.

Another important feature that distinguish WSNs is a somehow *reversed usage paradigm* with respect to traditional databases: there is no more a passive database and users who actively put queries on it, but queries are embedded into the sensors and actively send, possibly for long periods, *data streams* to a base station, where the application programs consume them. Moreover, sometimes queries can also specify actuation actions on the environment or on the measurement behaviour of the sensor; therefore each device can be, at the same time, a producer and a consumer of data.

Two phases must be considered in the life of a query in a WSN:

- the *query dissemination* phase, where queries are downloaded to the sensor's local memory. As to this phase, we must notice that the same sensor can be used for very different types of applications, with different usage modes. On the other hand, the syntactical structure of queries is fairly standard but for the presence or absence of some constructs and options [16]; *it is the context that makes the difference*. Therefore, starting from a standard skeleton, context can drive *query tailoring* in order to fulfill the application needs both as to the query text and to possible execution parameters, such as setting the sampling frequency or some alarm thresholds.
- the *result collection* phase, where the measurement data are fed back to applications. This phase is heavily affected by power management issues. As it was observed, once the battery is exhausted, the sensor dies since it is difficult or not worth to replace it; therefore energy sparing is one of the main optimization goals in query processing in WSNs. Since data transmission is more energy eager than local data processing, data preprocessing on a single sensor or within a sensors cluster is often needed and complex optimization techniques involving MAC, routing protocols and local processing have been proposed. For these reasons often applications are happy with approximate results, the bounds of which again depend on the *working context* of the application and are embedded in the downloaded query.

Last, but not least, sensors not only feed data directly to the application, but, as shown in Figure 12.1, they also provide numeric values for evaluating the context dimensions. Let us suppose that, during the ageing phase, the wine can be in three different *context situations*: `initial, middle, mature`; these situations are defined over three physical variables – temperature, acidity, activity – which are respectively sensed by thermometers, Ph-meters, and acoustic meters, and are subject to the following rules:

situation = `initial` :=
$\{(temperature < \ xx\ C)\ AND\ (acidity < \ Ph\ y)\ AND\ (noise < \ z\ db)\}$;
situation = `middle` :=
$\{(temperature > \ xx\ C)\ AND\ (acidity = \ Ph\ y)\ AND\ (noise > \ z\ db)\}$;
situation = `mature` :=
$\{(temperature < \ xx\ C)\ AND\ (acidity > Ph\ y)\ AND\ (noise < \ z\ db)\}$;

Therefore, data provided by sensors might directly affect the actual context through a feedback path, possibly inducing a context change. Such systems must be carefully designed and the sensor data must be cleaned and filtered in order to prevent the arising of possible instability situations.

In Chapter 18, a data management language for sensor data developed within the ART-DECO project, is presented, which deals with many of the mentioned functionalities.

12.6 Selecting Context, Query and Query Answering: A Web Portal

The context-aware knowledge querying concepts illustrated in the previous sections have been exploited through a case study in a wine production scenario.

Before going into the details of the specific study case, it is important to provide some general information on the Context-Aware Web Portal. First of all, it should not be seen as a mere container of information, rather it is an access point for creating knowledge by sharing and collecting information and experiences through a combination of contents, processes, technological and human resources. It is a framework for integrating information, people and processes across boundaries, and provides a secure unified access point in the form of a web-based user interface, designed to aggregate and personalize information through application-specific portlets. One feature of the web portal is the de-centralised content contribution and content management, which keeps the information always updated. Fundamental features of the web portal generated for the case study are:

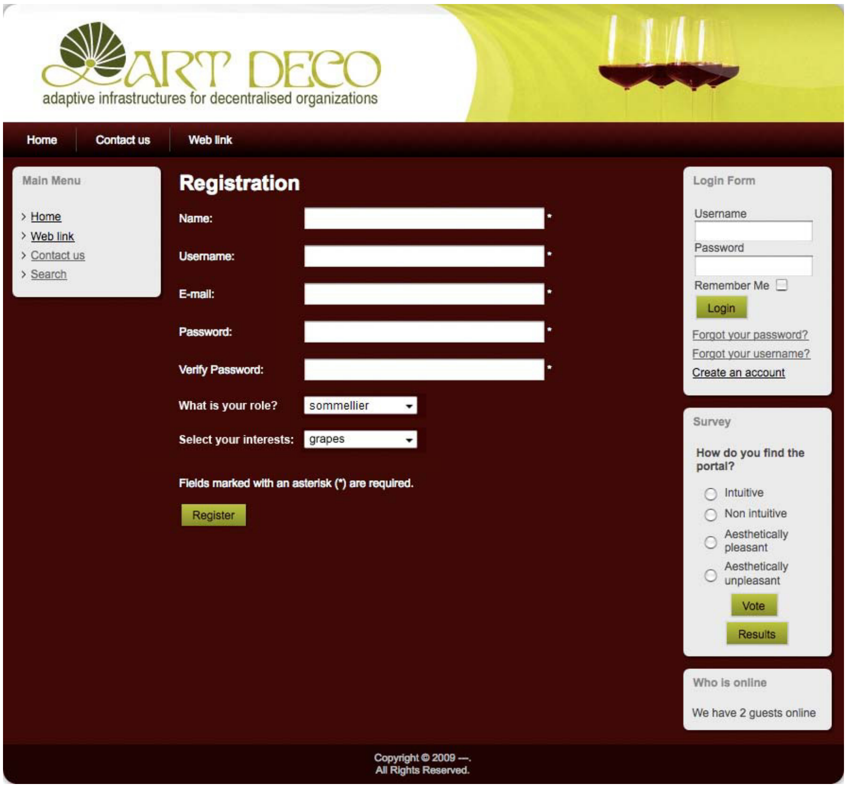
- *Single Sign-On* capabilities between the users and various other systems. As so, the user authenticates her/himself only once;
- *Integration*, the connection of functions and data from multiple systems into new components/portlets/web parts with an integrated navigation between these components;
- *Federation*, the integration of data coming from heterogeneous sources is transparent to the user;
- *Customization*, users can customize the look and feel of their environment. This aspect also refers to the ability to prioritize most appropriate content based on attributes of the user and metadata of the available content;
- *Personalization*, consists in matching content with the user. Based on a user profile, personalization uses rules to match the "services", or content, to the specific user;
- *Access Control*, it is possible to limit specific types of content and services users have access to, based on their role. These access rights may be provided by a portal administrator or by a provisioning process. Access control lists manage the mapping between portal content and services over the portal user base. This assures that each user has a narrower view over the large amount of information contained in the system, i.e. only the portion of information considered relevant

for that user, according to the context, can be accessed, reducing the risk of disorientation and avoiding having to dodge through quantities of unnecessary data, as the information that appears to the user is retrieved dynamically from the system's data sources.

The web portal communicates with the system architecture through web services, which filter the user's requests and break them down with respect to the single architectural components involved. This occurs independently of the type of information source, internal, external or coming directly from sensors used for monitoring the production processes as shown in Figure 12.1.

As previously mentioned, our case study application refers to a wine production context. Here, the contextual views are designed accordingly to the CDT, identifying different classes of users. Any user accessing the system for the first time is requested to register and during this process s/he is guided through a set of selections, conforming to the CDT dimensions, contributing to the determination of the "static" aspects of his/her active context (Figure 12.3).

Once the profile has been defined, each time s/he logs into the system, only the subset of information, relevant to the context, are shown (Figure 12.4). In this

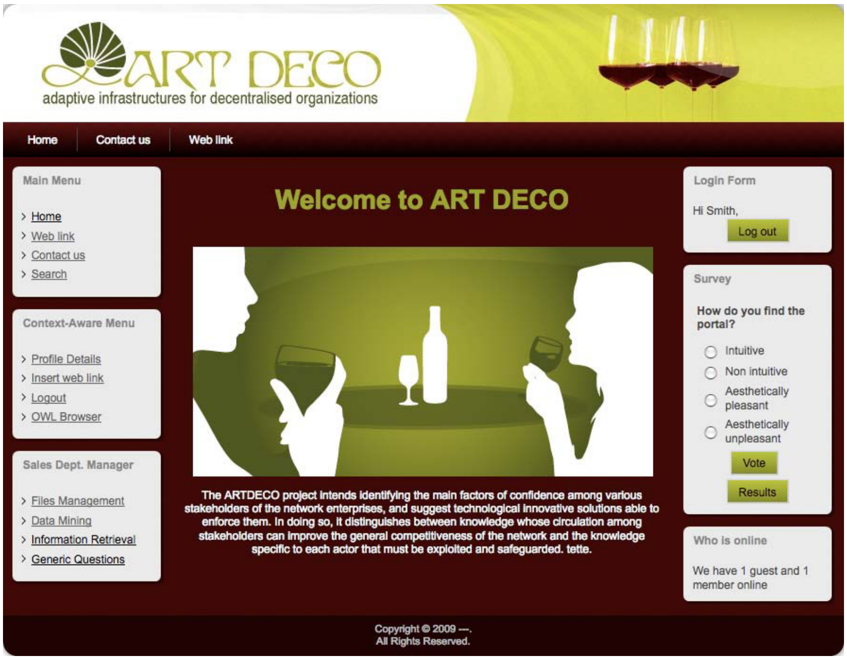


Registration

Name:	*
Username:	*
E-mail:	*
Password:	*
Verify Password:	*
What is your role?	sommellier
Select your interests:	grapes

Fields marked with an asterisk (*) are required.

Fig. 12.3 Access to the portal and context determination



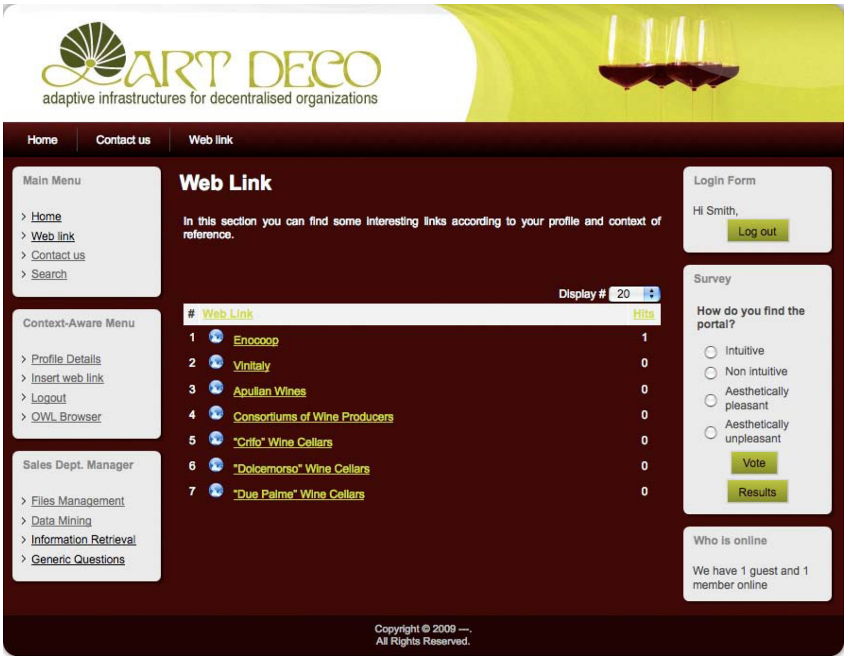
The ARTDECO project intends identifying the main factors of confidence among various stakeholders of the network enterprises, and suggest technological innovative solutions able to enforce them. In doing so, it distinguishes between knowledge whose circulation among stakeholders can improve the general competitiveness of the network and the knowledge specific to each actor that must be exploited and safeguarded. tette.

Fig. 12.4 User authentication and context-aware interface generated for `role=sales dept.manager`

scenario the user can carry out a set of actions leading to different results according to the context.

The interface contains various parts. The "Main Menu" section includes, among others, a generic `web-link` feature common to all roles. It suggests a set of possible links that may be of interest for the specific user. So, although the feature is common to all profiles, the contents differ from one another. Also, each user may upload a document or a URL through a dedicated interface. The documents and links are made available to other users with the same context characteristics and therefore likely to have the same interests (Figure 12.5). A survey has also been structured to acquire the users' viewpoint on usability aspects. Any user is free to answer the questionnaire; only the first question of the survey is shown here, but once the user answers, the entire survey is displayed in a new page.

The specific parts of the portal relate to the "Context Aware Menu" and the "User Profile" in the lower left part. The first contains features for customizing and personalizing one's profile. Here a user can upload a Web Link which is classified as relevant for that certain context. The latter is specific of the role represented. Indeed, the header of the box is one's own role. We will comment the most relevant features of the specific parts related to the wine production scenario. A first functionality is the information retrieval. This is exploited through a semi-structured approach where s/he selects the information in a dedicated interface. A query is processed,


ART DECO
adaptive infrastructures for decentralised organizations

Home | Contact us | Web link

Main Menu
- Home
- Web link
- Contact us
- Search

Context-Aware Menu
- Profile Details
- Insert web link
- Logout
- OWL Browser

Sales Dept. Manager
- Files Management
- Data Mining
- Information Retrieval
- Generic Questions

Web Link

In this section you can find some interesting links according to your profile and context of reference.

Display # 20

#	Web Link	Hits
1	Enocoop	1
2	Vinitaly	0
3	Apulian Wines	0
4	Consortiums of Wine Producers	0
5	"Crifo" Wine Cellars	0
6	"Dolcemorso" Wine Cellars	0
7	"Due Palme" Wine Cellars	0

Login Form
Hi Smith,
Log out

Survey
How do you find the portal?
- Intuitive
- Non intuitive
- Aesthetically pleasant
- Aesthetically unpleasant

Vote
Results

Who is online
We have 1 guest and 1 member online

Copyright © 2009 —.
All Rights Reserved.

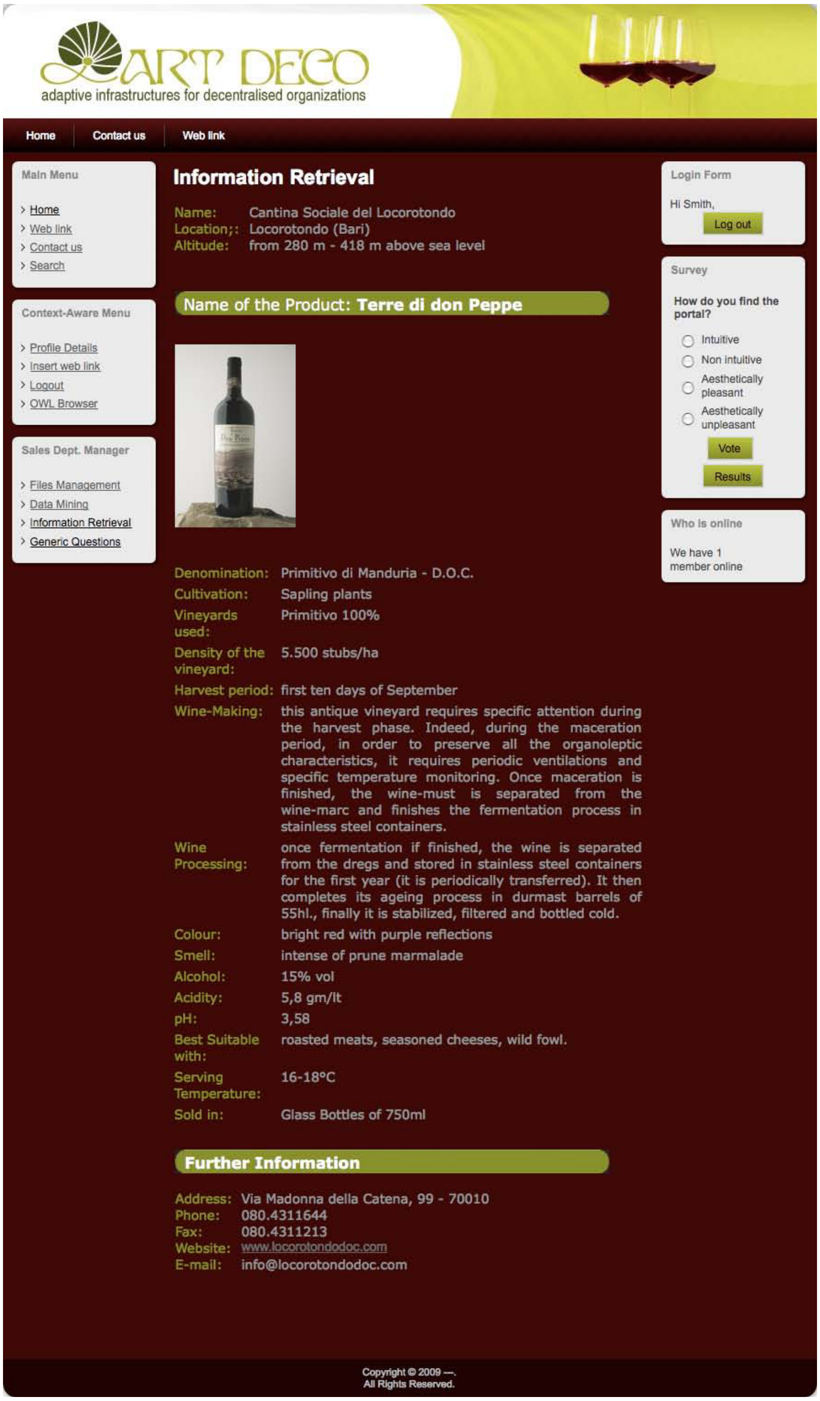
Fig. 12.5 Web-Links of interest for the `sales dept.manager`role

and data is extracted from the data sources. For example, suppose that a sales department manager wants to know the wines that have been top sellers in a specific year (i.e., `dept` = `SalesDept`, `time` = `year`(`$yearID` =`2010`), and `interest` = `wine`), the result is shown as a list of wines that fulfill the interrogation (Figure 12.6).

The underlying Context-Aware Application Server provides, as the answer, only the portion of data corresponding to the user's active context. So, if on the other hand an oenologist were to ask the same information (`time` = `year`(`$yearID` =`2010`) and `interest` = `wine`) but with a different role (`role` = `oenologist`), data on the cultivation and harvesting, collected from the sensors, would also appear, because the role is different, and therefore the view on the data changes. To summarize, in accordance to the previous sections, the web portal shows different portions of data to different users according to the combination of context dimensions, given a specific role.

As final application of the case study, it is worth mentioning that a user can also ask generic questions guided through a dedicated interface where s/he selects the parameters and the query is dynamically structured and sent to the system architecture components. Some examples of queries, based on the CDT, may be:

- Average time needed to provide an invoice [supplierID];
- Average temperature in cellar areas during a [day]/[month]/[year]
- Which are the most trendy wine origins/wine-food pairings/glasses/colours during the period [start]-[end]



Information Retrieval

Name:	Cantina Sociale del Locorotondo
Location;:	Locorotondo (Bari)
Altitude:	from 280 m - 418 m above sea level

Name of the Product: **Terre di don Peppe**

Denomination:	Primitivo di Manduria - D.O.C.
Cultivation:	Sapling plants
Vineyards used:	Primitivo 100%
Density of the vineyard:	5.500 stubs/ha
Harvest period:	first ten days of September
Wine-Making:	this antique vineyard requires specific attention during the harvest phase. Indeed, during the maceration period, in order to preserve all the organoleptic characteristics, it requires periodic ventilations and specific temperature monitoring. Once maceration is finished, the wine-must is separated from the wine-marc and finishes the fermentation process in stainless steel containers.
Wine Processing:	once fermentation if finished, the wine is separated from the dregs and stored in stainless steel containers for the first year (it is periodically transferred). It then completes its ageing process in durmast barrels of 55hl., finally it is stabilized, filtered and bottled cold.
Colour:	bright red with purple reflections
Smell:	intense of prune marmalade
Alcohol:	15% vol
Acidity:	5,8 gm/lt
pH:	3,58
Best Suitable with:	roasted meats, seasoned cheeses, wild fowl.
Serving Temperature:	16-18°C
Sold in:	Glass Bottles of 750ml

Further Information

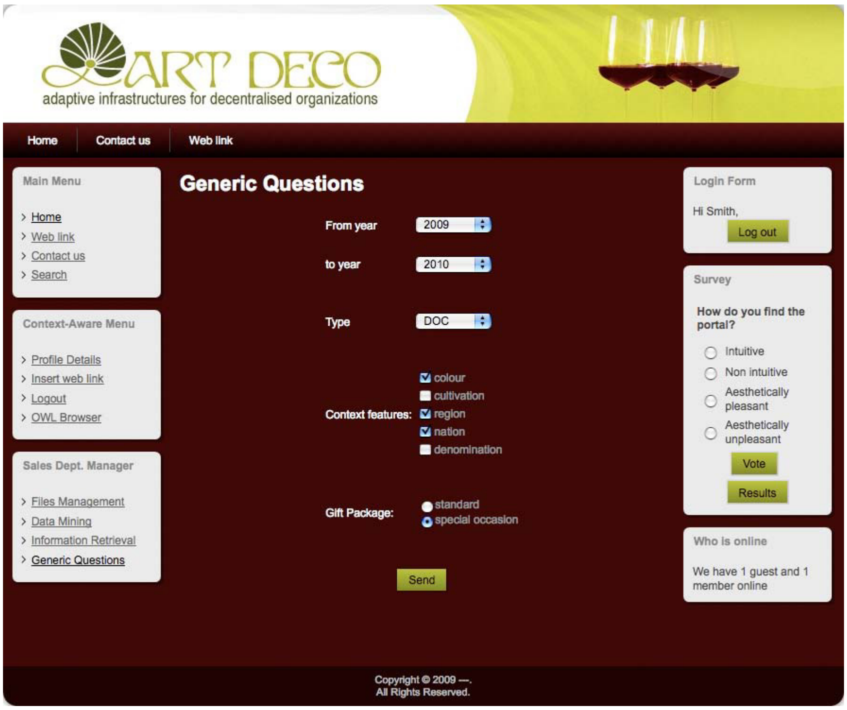
Address:	Via Madonna della Catena, 99 - 70010
Phone:	080.4311644
Fax:	080.4311213
Website:	www.locorotondodoc.com
E-mail:	info@locorotondodoc.com

Copyright © 2009 —.
All Rights Reserved.

Fig. 12.6 Context-Aware Information Retrieval

- Which wines were drank by the various trend setters during [year]
- Which are the most trendy gift package ideas for DOC red wines produced in Sicily in years [start]-[end]?

These queries are generated from an interface which varies according to the context. Moreover, Figure 12.7 shows an example of the interface for the last generic question of the list.



Generic Questions

From year 2009

to year 2010

Type DOC

Context features: ☑ colour ☐ cultivation ☑ region ☑ nation ☐ denomination

Gift Package: standard [✓ special occasion]

Hi Smith,

How do you find the portal?

Intuitive / Non intuitive / Aesthetically pleasant / Aesthetically unpleasant

We have 1 guest and 1 member online

Fig. 12.7 Example of Generic Question

As it can be seen, the entire process of query building, information search and retrieval, and results is completely transparent to the user, who has no perception of which system components the data comes from, nor how his/her view differs from or is similar to other ones. In this way the web portal offers a systematic manner for dynamically extracting a specific context-aware view over the entire system.

12.7 Related Work

Context modelling refers to the representation of a user and his/her context at the level of a system profile. Adequate modelling of the user and his/her context is the basis for every form of personalization support. There are different lines along which a context model should be analysed in order to be adopted: (a) the modelled aspects; the set of context dimensions managed by the model (e.g., time, location, situation, ...); (b) the representation formalism (graph-based, logical, etc.); (c) the language (e.g., XML). These choices heavily influence context manageability and dynamicity.

Sophisticated and general context models have been proposed, to support context aware applications which use them to (a) adapt interfaces [9], (b) tailor the set of application-relevant data [3], (c) increase the precision of information retrieval [18], (d) discover services [14], (e) make the user interaction implicit [12], or (f) build smart environments [10].

In the ART-DECO perspective, we are interested to data-oriented approaches, rather than methodologies and systems referring to a general notion of user and context. Among the approaches devoted to selecting subsets of data based on the notion of context, we count the Context-Relational model (CR) [15], which proposes the use of relational tables having an additional dimension, accommodating different versions of the same information, to be made available according to the current context. Analogously, in [19], context is modelled as a set of multidimensional attributes, data cubes are used to store the dependencies between context-dependent preferences and database relations, and OLAP techniques are adopted for processing context-aware queries. In [7], [21] an approach is presented in which context is used to tailor data, coming from multiple information sources, in order to discard information not relevant to the current situation, like in the ART-DECO scenario.

The interested reader can refer to [20, 1, 4] for comprehensive surveys on context models.

12.8 Conclusions

In this chapter, we showed how the scenario of a a networked enterprise is rich of information and data to be shared among different users, each one characterized by a specific role and interests. Furthermore, other distinguishing aspects may affect the portion of such broad information the user is interested in. As a consequence, the adoption of a contextual approach to data access is particularly interesting, to limit information overhead and noise. In this chapter we have presented a flexible contextual model supporting data tailoring in a rich, heterogeneous data environment, where both internal and external sources are available. Such a model is used as the starting point for the application of a methodology for the association of a portion of data deemed interesting for each possible context the user may be in. This activity is performed at design time, when the application scenario is envisioned, establishing the relation between the aspects characterizing the context, the possible significant contexts and relevant information that should be associated with each of them. At run time, the user's active context is exploited to effectively perform a dynamic tailoring of the available data, to retrieve only the significant information.

Within the ART-DECO project, the methodology has been developed and applied to the wine production case study, here presented, where the application providing the context-aware access to data is a web portal, developed to interact with a context-aware application server to support the specification of the user's active context and its exploitation to access filtered data.

References

1. Baldauf, M., Dustdar, S., Rosenberg, F.: A survey on context&#45;aware systems. Int. J. Ad Hoc Ubiquitous Computing 2(4), 263–277 (2007)
2. Bolchini, C., Curino, C., Quintarelli, E., Schreiber, F.A., Tanca, L.: Context information for knowledge reshaping. Intl. Journal of Web Engineering and Technology 5(1), 88–103 (2009)
3. Bolchini, C., Curino, C., Schreiber, F.A., Tanca, L.: Context integration for mobile data tailoring. In: Proc. 7th IEEE/ACM Int. Conf. on Mobile Data Management, p. 5 (2006)
4. Bolchini, C., Curino, C.A., Quintarelli, E., Schreiber, F.A., Tanca, L.: A data-oriented survey of context models. SIGMOD Record 36(4), 19–26 (2007)
5. Bolchini, C., Quintarelli, E.: Context-Driven Data Filtering: A Methodology. In: Meersman, R., Tari, Z., Herrero, P. (eds.) OTM 2006 Workshops, Part II. LNCS, vol. 4278, pp. 1986–1995. Springer, Heidelberg (2006)
6. Bolchini, C., Quintarelli, E., Rossato, R.: Relational Data Tailoring Through View Composition. In: Parent, C., Schewe, K.-D., Storey, V.C., Thalheim, B. (eds.) ER 2007. LNCS, vol. 4801, pp. 149–164. Springer, Heidelberg (2007)
7. Bolchini, C., Schreiber, F.A., Tanca, L.: A methodology for very small database design. Information Systems 32(1), 61–82 (2007)
8. Carroll, J.J., Klyne, G.: Resource description framework (RDF): Concepts and abstract syntax, W3C recommendation. Technical report, W3C (2004)
9. De Virgilio, R., Torlone, R., Houben, G.-J.: A rule-based approach to content delivery adaptation in web information systems. In: Proc. 7th IEEE/ACM Int. Conf. on Mobile Data Management, p. 21 (2006)
10. Dey, A.K., Sohn, T., Streng, S., Kodama, J.: iCAP: Interactive Prototyping of Context-Aware Applications. In: Fishkin, K.P., Schiele, B., Nixon, P., Quigley, A. (eds.) PERVASIVE 2006. LNCS, vol. 3968, pp. 254–271. Springer, Heidelberg (2006)
11. Franconi, E.: Conceptual Schemas and Ontologies for Database Access: Myths and Challenges. In: Parent, C., Schewe, K.-D., Storey, V.C., Thalheim, B. (eds.) ER 2007. LNCS, vol. 4801, p. 22. Springer, Heidelberg (2007)
12. Petrelli, D., Not, E., Strapparava, C., Stock, O., Zancanaro, M.: Modeling context is like taking pictures. In: Proc. of the Workshop "The What, Who, Where, When, Why and How of Context-Awareness" in CHI 2000 (2000)
13. Prud'hommeaux, E., Seaborne, A.: SPARQL query language for RDF. Technical report, W3C (2007)
14. Raverdy, P.-G., Riva, O., de La Chapelle, A., Chibout, R., Issarny, V.: Efficient context-aware service discovery in multi-protocol pervasive environments. In: Proc. Intl Conf. on Mobile Data Management, p. 3 (2006)
15. Roussos, Y., Stavrakas, Y., Pavlaki, V.: Towards a context-aware relational model. In: Proc. Context Representation and Reasoning - CRR 2005, pp. 7.1–7.12 (2005)
16. Schreiber, F.A.: Automatic generation of sensor queries in a wsn for environmental monitoring. In: Proc. 4th Int. ISCRAM Conference, Delft (NL), pp. 245–254 (2007)

17. Schreiber, G., Dean, M.: OWL web ontology language reference, W3C recommendation. Technical report, W3C (2004)
18. Shen, X., Tan, B., Zhai, C.: Context-sensitive information retrieval using implicit feedback. In: Proc. Int. Conf. on Research and Development in Information Retrieval, pp. 43–50 (2005)
19. Stefanidis, K., Pitoura, E., Vassiliadis, P.: A context-aware preference database system. Journal of Pervasive Computing & Comm. 3(4), 439–460 (2007)
20. Strang, T., Linnhoff-Popien, C.: A context modeling survey. In: Workshop on Advanced Context Modelling, Reasoning and Management at UbiComp (2004)
21. Tanca, L.: Context-based data tailoring for mobile users. In: Proc. BTW 2007, pp. 282–295 (2007)