Technical Issues in the Development of Knowledge-Based Services for the Semantic Web*

Carlos Pedrinaci[1] and Tim Smithers[1] and Amaia Bernaras[2]

[1] Parque Tecnológico de San Sebastián. Paseo Mikeletegi, 53 20009 San Sebastián, Spain. cpedrinaci@computer.org, tsmithers@acm.org
[2] Idom. Paseo de los Olmos, 14 Bidebieta I 20016 San Sebastián, Spain. abernaras@idom.es

Abstract. The Semantic Web aims to extend the current Web with formal semantics in order to improve how users experience the Web, by ameliorating current activities and supporting the automation of some others. So far, current Semantic Web prototypes mostly aim at collecting and exposing information. Still, a semantic layer can support applying Knowledge-Based Systems techniques to the development of brand-new fully-fledged Knowledge-Based Services for the Web. In this paper, we present the technical issues that have to be faced in the development of such a kind of application by presenting the Online Design of Events Application: a Semantic Web-based design support system that assists event organisers in the process of preparing events such as workshops and conferences, by effectively reasoning over an inter-organisational process accross the Web.

1 Introduction

In recent years, the Semantic Web [1] and the technologies it builds upon have received a lot of attention from academic and industrial research. These efforts have mainly been reflected as a set of knowledge representation languages, formats and libraries that manipulate them. So far, prototype systems developed upon these technologies mostly aim at collecting information and offering it, lacking more advanced capabilities that a semantic layer can support.

Semantic Web technologies pave the way for improving human-computer interaction, increasing the support and automation of some activities, ameliorating the outcome of some others, and can lead to the generation of new business opportunities. In fact, the Semantic Web is largely considered as a prime enabler of future forms of collaborative eBusiness.

Formal semantics can bring brand-new opportunities for applying knowledge-based reasoning techniques to the Web. However, applying these technologies to inter and even intra-organisational processes presents important challenges. It

* This work has been partially supported by the European Commission, as project OBELIX (IST-2001-33144)

requires the application of an important body of knowledge about current state-of-the-art technologies and their integration with engineering practices.

In this paper, we present the Online Design of Events Application (ODEA), an exploratory case study on the development of advanced Knowledge-Based Services over the Web. ODEA is a Semantic Web-based design support system that assists event organisers in the process of preparing events such as workshops and conferences, by effectively reasoning over an inter-organisational process engaging diverse agents communicated accross the Web.

The remainder of this paper is organised as follows. We first present the conceptual understanding of events organisation which underlies the application we have implemented. We next describe our conceptual approach to supporting events organisers by building our application on top of our Opportunistic Reasoning Platform. Finally, we tackle some technical aspects of the development of ODEA.

2 ODEA: a Semantic Web-based Design Support System

So far, San Sebastian Technology Park (PTSS) has hosted numerous events. The experience shows that organising events like conferences and meetings usually involve an important load of work. It usually requires preparing a reasonable schedule and booking the needed furniture and technical equipment. It often involves contacting external providers for renting some equipment, booking additional facilities (e.g., a catering service, a translation service) and it may even involve accommodating the participants.

The majority of the tasks involved in any event organisation have to be repeated over and over again for every specific situation. Moreover, in many cases, due to resource allocation, schedule changes or other difficulties, part or even the whole event preparation has to be reworked. For example, depending on the dates and the number of participants, it may happen that it is impossible to obtain all the equipment required. Such a situation could lead to a schedule change thus invalidating previous resources allocations, service bookings, etc.

In summary, organising events involves many tasks that can, and certainly should, be automated. To this end we have developed the Online Design of Events Application (ODEA): a Semantic Web-based system, that aims to support PTSS employees in the process of designing events where the service provided should be configured and customised to the needs of the type of event and client. In this application, events organisation is understood as a knowledge-intensive activity involving, not only PTSS, but also other providers, such as audiovisual equipment providers, restaurants and hotels, in a complex value chain that is needed when hosting events. Therefore, collaboration between different entities plays an important role in ODEA offering new opportunities for smart collaborative eBusiness.

We have engineered ODEA based on the $K_L D^E_{V1}$ [2, 3]. The $K_L D^E_{V1}$ is a Knowledge Level [4] theory of designing which intends to provide a theoretical support for designing, that could offer as well an appropriate framework for the

Knowledge Engineering and construction of Knowledge-Based design systems. The theory attempts to define the necessary and sufficient kinds of knowledge involved in all and any kind of designing, the roles they play and the relationships between them. The $K_L D^E_{V1}$ characterises designing as the "the exploration of the problems that can be devised whose solutions can be shown to satisfy the needs or desires that motivate the designing." [5].

Based on the empirical understanding of our specific domain and informed by the $K_L D^E_{V1}$, we have created a model for organising events, namely the Events Design Process Model shown in Figure 1. The main role of ODEA is to assist the user in the process of designing an event. Thus, its main concern is to effectively support event designers through a smooth exploration of the possible event designs by guiding the designing process, assisting the user in the specification of configuration problems [6], solving these problems, contacting external providers and maintaining a coherent state of the overall design.

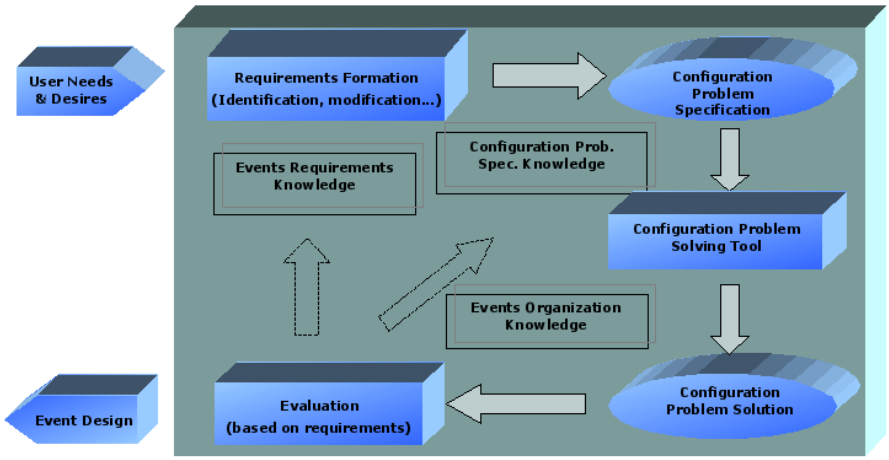

Fig. 1. ODEA's Design Process Model.

Starting with the needs and desires of a client, the Event Design Process Model supports the identification of a set of requirements that, when satisfied, result in an event design acceptable to the client. On the basis of the requirements obtained, the system supports the development of well-formed configuration problem specifications that are essentially attempts to operationalise some or all of the requirements previously identified. The solutions that are obtained are evaluated with respect to the requirements. The outcome of this evaluation then leads either to the identification of further or modified requirements, a different operationalisation, or to a final design.

3 Design of ODEA

Supporting an event designer in the iterative, incremental, and opportunistic exploration of the possible event designs, requires smoothly enabling a mixed initiative between the user and the system, which performs the tedious tasks on behalf of the designer, yet allowing him or her to guide the overall design. Further, the system must support the seamless integration of external providers into the overall designing process. In order to support ODEA we have built it on top of our Opportunistic Reasoning Platform. Figure 2 depicts how the design support system has been supported by our reasoning platform, and identifies the main components of the system.

The core of our reasoning platform, which is shown in the bottom left of Figure 2, is our own implementation of a Blackboard Framework [7], that is, a skeletal implementation of the Blackboard Model [7]. The Blackboard Model is a generic, widely applicable and well-known reasoning model that was introduced as a means to overcome the difficulties of traditional Knowledge-Based Systems development techniques. It is often presented using the metaphor of a group of persons trying to put together a jigsaw puzzle, incrementally (one at a time) and opportunistically (as they see a new piece can fall into place). The fundamental philosophy of this problem-solving model establishes that experts, also known as Knowledge Sources (KSs) in the blackboard literature, do not communicate with each other directly, instead all the interactions strictly happen through modifications on the shared blackboard.

In ODEA we adopt this reasoning model for it exhibits and honours a good set of Software and Knowledge Engineering best practices, such as software and knowledge modularity, extensibilitity and adaptability. Still, our Blackboard Framework presents an important novelty with respect to traditional implementations. In our framework Knowledge Sources have been externalised in order to adapt them to the Web. Doing so allows us to better support reasoning over the Web, by benefiting from Knowledge Sources expertise in the interaction with external agents (i.e., the user and external providers), thanks to the use of ontologies as the *lingua franca* and Web Services to interact with remote systems.

Figure 2 depicts the different Knowledge Sources that have been developed, the mechanisms applied in order to integrate the Knowledge Base, the inference engines the Knowledge Sources have used, and the means by which external agents have been incorporated into ODEA. In the remainder of this section we will describe in detail these different components, paying first a special attention to the Knowledge Sources developed and how they contribute to supporting the event designer. Finally, we will tackle technical implementation details and the issues that had to be faced.

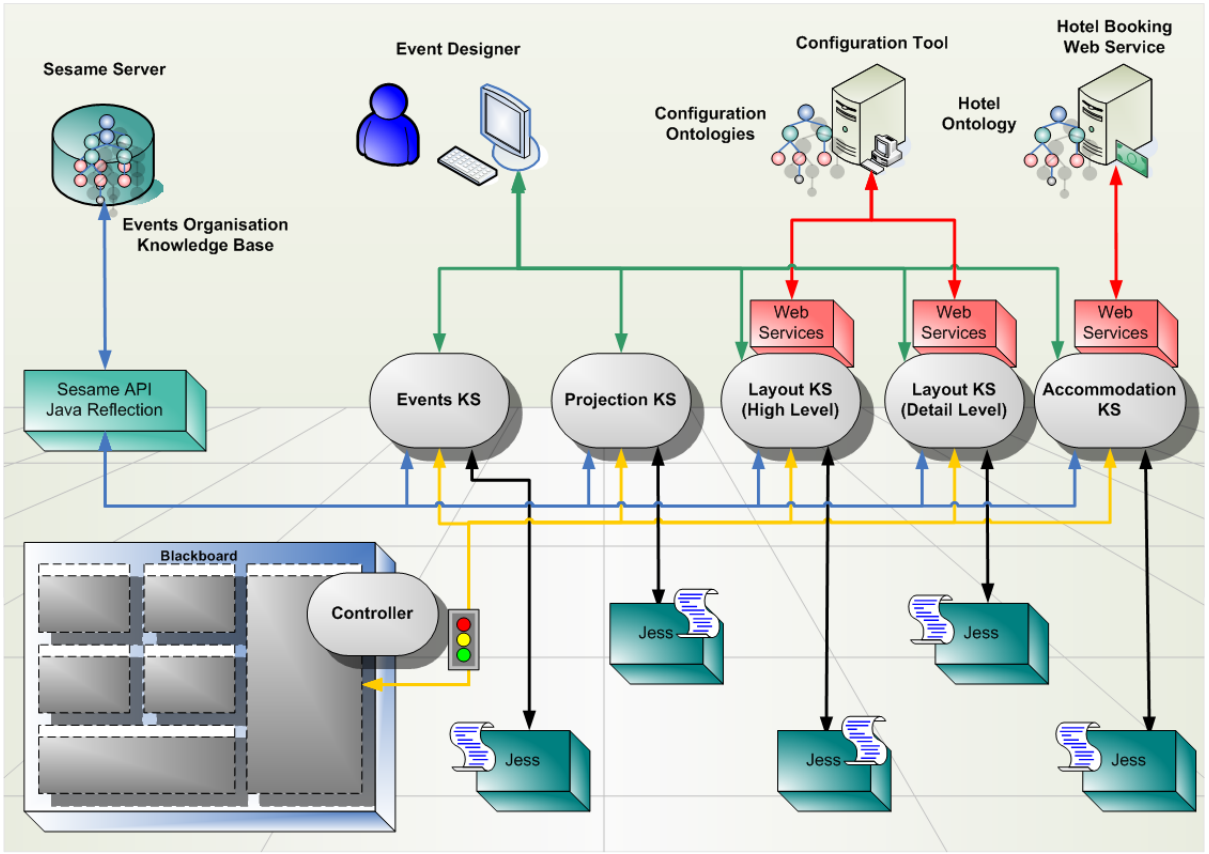

Fig. 2. Architecture of ODEA based on our Opportunistic Reasoning Platform.

3.1 The Events Organisation Knowledge Base

Appropriately organising events requires applying an heterogeneous body of knowledge. This includes general knowledge about the different kinds of events, knowledge about additional services like catering, projection and accomodation, etc. To support the system in this very activities, ODEA is informed by a Knowledge Based expressed in terms of three ontologies [8]:

Events Types Ontology The Events Types Ontology defines the different types of events such as Meeting, Conference or Exhibition, together with some general information like the dates, the attendants or the lecturer. This ontology has been defined in generic terms in order to favour its reusability, but specific enough so as to be useful for ODEA in supporting PTSS event designing. It is therefore thought to be reused in other systems for describing events without any other burden associated to their organisation.

Events Equipment Ontology The Events Equipment Ontology defines the resources required for organising an event. This ontology is, to some extent more oriented towards ODEA even though its conceptualisation could partly be reused in other applications. This ontology characterises the equipment as being furniture, such as tables and chairs, or technical which comprises computers, videoprojectors, screens, etc. The different kinds of equipment have further been detailed in terms of their brand, model, price, size, etc. The equipment currently available in PTSS has been expressed in terms of this ontology and stored in a Knowledge Base in order to be used at runtime by the design support system.

Hotel Accommodation Ontology The Hotel Accommodation Ontology defines the concepts regarding the accommodation and complementary services usually offered by hotels. It therefore defines rooms, which are further characterised as single, double or suite, as well as it defines the concept reservation and additional services such as room facilities (e.g., TV), the provision of meals (e.g., half board), etc. This ontology is intended to be used as a shared conceptualisation with hotels systems and therefore supports ODEA in the task of checking and booking rooms automatically.

Ontologies streamline the effective integration of the different experts composing the system, by establishing a common "language" between them. Thus, ontologies provide a convenient solution to a typical difficulty developers of Blackboard Systems have to face, concerning the representation of sharable yet expressive and extensible conceptual schemas to be shared among the Knowledge Sources [7]. In a similar way, as we will see next, ontologies have also played an important role in supporting the interaction with external agents.

3.2 Knowledge Sources

The Blackboard Model establishes that a set of experts, known as Knowledge Sources, collaborate in the overall problem-solving. ODEA supports the user

in the process of designing events by means of five Knowledge Sources (see Figure 2).

In ODEA Knowledge Sources are hierarchically organised in different levels of abstraction over the event design process. The hierarchy establishes that the Knowledge Sources of a particular level can access the data being handled by the Knowledge Sources situated directly below. For instance, the *Events KS* is aware of the work being done by the *Projection KS*, the *Accommodation KS*, and *Layout KSs*. This mechanism supports a smooth an efficient flow of the information between the levels of abstraction.

Events Knowledge Source The Events Knowledge Source encapsulates general expertise about events and their organisation. In particular, the different kinds of events have been modelled together with their specific characteristics. These characteristics are derived from the study previously performed and shown in [9], which is based on historical data of the events hold in PTSS. Therefore, it is important to note that even though some data has been generalised in order to favour reusability, the majority of the information is influenced by PTSS context.

For example, a meeting is characterised as an event for supporting discussion. Thus, in order to facilitate discussion, participants are usually placed in an "O-Layout" (i.e., a placement forming an "O") or an "U-Layout" in the cases where some sort of projection takes place. Obviously no meeting can take place with less than two participants. The same way meetings with more than fifty participants are unlikely to be successful. The usual number of participants have been determined as varying between five and forty. In addition to these general information, resources usage has also been analysed and classified into possible and recommended based on the type of event. For instance, it is usual that in this kind of event, participants use computers and flipcharts which do also impose other requirements on electricity sockets, internet connectivity, etc.

Besides, encapsulating general knowledge about events designing, the Events KS plays an integrating role. In fact, even though the designing expertise is partitionned into several Knowledge Sources, there will always remain relationships we will need to address. For instance, once we have organised the people we must check whether the organisation is compatible with the projection service, since having a projection service when the participants are organised in an "O-Layout" would prevent some of them from seeing the projections. In ODEA the Events KS is the top-level KS (see Figure 2) and holds the knowledge concerning the integration of the solutions generated by the Layout, Projection and Accommodation KSs. It is worth noting, that such a kind of integration is effectively and efficiently supported by the shared conceptualisations—the ontologies—and by the hierarchical organisation of the KSs.

Projection Knowledge Source The current implementation of ODEA supports the event designer in the task of determining whether a particular projection service can be set-up or not and how it could be furnished, thanks to the

Projection KS. This KS is aware of the internal details associated to offering projection solutions during an event and has access to the hardware available together with their characteristics.

The Projection KS offers to the event designer the possibility to browse, query, retrieve and reserve projection hardware for the event being designed. To that end, the Projection KS provides a user interface based upon the concepts and attributes defined in the Events Equipment Ontology previously presented. Through this user interface, the event designer can retrieve the available hardware, based on any combination of criteria over the equipment attributes, in ontological terms which are presumably close to the designer understanding. At runtime, based on the selected criteria, the Projection KS generates the appropriate query, obtains the suitable hardware from the Knowledge Base, and presents it to the designer who can then choose his or her preferred one.

This Knowledge Source is perhaps the one that better shows the benefits we obtain from supporting the user interaction thanks to being informed by ontologies. Establishing Knowledge Sources as the entry point for the event designer, has supported the creation of a semantics-based interface which benefits from the expertise encapsulated in order to better adapt it to the user. Moreover, because the interaction with the event designer is already performed in terms of the Events Equipment Ontology, the input can directly take part into further reasoning processes. This particular Knowledge Source therefore illustrates one of the main aims of the Semantic Web, that is, "better enabling computers and people to work in cooperation" [1].

Accommodation Knowledge Source Organising events often implies reserving hotel rooms for some of the participants. We have therefore decided to use hotel reservation as an example for exploring how to support reasoning over the Web in collaboration with external agents. Further, this Knowledge Source illustrates how making an effective use of Semantic Web and Web Services technologies can offer new collaborative eBusiness opportunities when they are coupled with the appropriate reasoning software infrastructure.

In the Online Design of Events Application we have integrated such a capability via the Accommodation KS so that, at runtime, the designer can automatically contact a hotel, check the availability of rooms and finally book the hotel rooms desired. This KS encapsulates the insights associated to hotel rooms reservations. It is aware of the different kinds of rooms (single room, double room) and the enhancing services that can usually be offered by hotels. This conceptualisation of the hotels reservation domain—the Hotel Accommodation Ontology—is shared with a remote hotel reservation system. During an event design, Web Services technologies (i.e., WSDL and SOAP) support the effective communication with the remote hotel service and the shared conceptualisations (i.e., the ontology) leverage the interaction to a semantic level thus supporting the integration of the hotel service into the reasoning processes and paving the way for further integration of other hotel systems.

Layout Knowledge Sources We have previously raised the importance for an appropriate organisation of the participants during a meeting. In general, people organisation is an important aspect in any kind of event. Therefore ODEA supports an event designer in the task of appropriately setting-up the people, that is, setting up some tables in a suitable way, so that people can be disposed as desired.

We currently consider four different layouts, "O", "U", "Class-room" and "Theatre". The first layout organises the tables in a way such that the people form a rectangle or a circle. It is suitable for relatively small sized events and is not compatible with a projection service. The second one is very similar but leaves one of the borders free, so that it can be used for projection. The Class-room layout sets the people in different rows just like in traditional class rooms. This is often the preferred layout for Conferences and Lectures. Last, the Theatre layout is very similar but better suited for a large number of participants given that the rows are staggered.

While people organisation is often a trivial task, in many situations this is far from reality, not to mention the potential savings that can be obtained by optimising the number of tables used, by minimising the transfers between meeting rooms, etc. In ODEA we have used people organisation as an illustrative example showing how Problem-Solving Methods can be seamlessly integrated into problem-solving activities over the Web.

Participants organisation in ODEA is characterised as a configuration problem [6] where, given the number of participants, the set of tables available and the desired layout, we have to find a suitable configuration. In order to support organising people, we have integrated a generic configuration problem-solving tool developed by Labein in the course of the OBELIX project [10]. The configuration tool is a Problem-Solving Method implementation [11] that encapsulates the expertise for solving configuration problems independently from the application domain. This expertise includes the sequences of the different inference steps that have to be performed as well as the different kinds of knowledge required during the problem-solving process.

The configuration tool is informed by the Configuration Ontologies which establish the concepts and relationships required for solving and specifying configuration problems. There are three Configuration Ontologies:

Components Ontology The Components Ontology represents the different concepts and relationships necessary to define the static knowledge of a specific domain. This knowledge is expressed in terms of components having a set of ports and described by additional properties. Components can be connected to other components via associations, or via connections between compatible ports.

Constraints Ontology This ontology supports defining the constraints that must be applied over the components.

Problem Specification Ontology This ontology allows applications to define concrete problem specifications with some optimal criteria if necessary.

The configuration tool we have used differentiates two levels of detail for configuration problems. *High level* configuration problems are only concerned with determining whether several elements can be put together. These are usually known as *association* or *aggregation* problems. *Detail level* configuration problems do also deal with how these elements are put together, how they are *connected*. These are known as *connection* problems.

In a very similar way to what is described in [12, 13], ODEA integrates the configuration tool by mapping the Events Equipment Ontology to the Configuration Ontologies. Thus, we have described the tables according to the Components Ontology, that is, we have defined their ports with their respective connectivities and we have included additional parameters such as the tables dimensions. This definition has been stored in the so-called Events Components Ontology. Moreover, we have defined constraints according to the Constraints Ontology, in order to organise the tables in the different layouts, or to limit the number of persons per table. In this case the definitions have been represented in the Events Constraints Ontology.

Together with these definitions an intermediate module, which can be seen as an "*adapter*" in terms of [14], provides the means for interacting with the configuration tool through simple Java methods. Problem specifications are generated thanks to the Problem Specification API allowing application developers to generate problem specifications at runtime. The same way, applications can browse the solutions by using the Problem Solution API (see Figure 3). This way, at runtime, the Layout Knowledge Sources may generate a problem specification to be solved by the Configuration Tool. Once the tool has returned the solutions (if any) the Knowledge Source can interpret the results and update the blackboard accordingly.

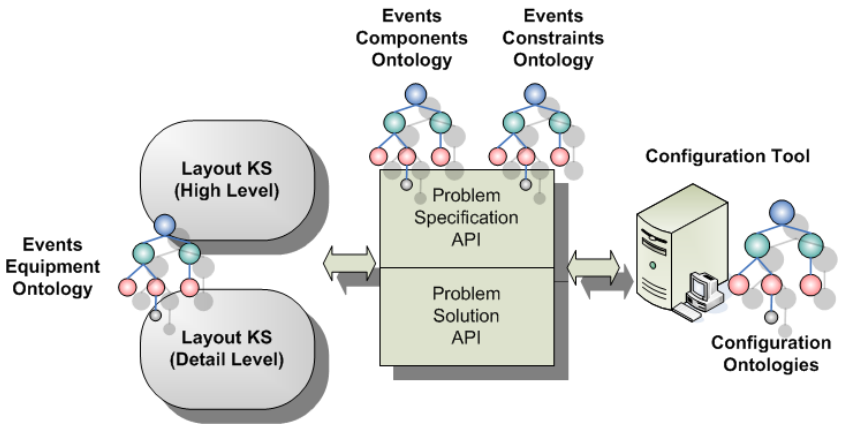

Fig. 3. Configuration Tool Integration.

In order to smoothly integrate the event designer as well as to reduce the computational resources required for solving the configuration problems, we have decomposed the tables organisation process into two different steps. The first step is in charge of determining whether it is possible or not to find a solution with the available components. This is achieved through the resolution of a high level configuration problem whose outcome are the possible associations. The second, step consists on solving the detail level configuration problem for the solutions retrieved in the first step. This second step thus determines how the required tables must be connected.

The Online Design of Events Application is supported by two Knowledge Sources to achieve such a decomposition: the Layout KS (High Level) and the Layout KS (Detail Level) (see Figure 3). The former is in charge of generating and interpreting high level configuration problems for organising the tables. The latter takes care of refining particular high level solutions by constructing and interpreting detail level configuration problems.

At runtime, the system first determines the possible associations of tables that could fulfil the user requirements. The solutions are then interpreted, and presented to the event designer. As determined by the Events Design Process Model (see Figure 1), the evaluation of these solutions can lead to further detailing or changing the problem specification. Whenever further specification is desired, the Layout KS (Detail Level) generates a detail level configuration problem, which is again solved and presented to the user. The event designer is then given the possibility to select the solution that he or she prefers, and to establish it as the tables organisation for the event being designed.

In this section we have presented a high-level view of the Online Design of Events Application. We have given a brief overview of the overall architecture of the system which builds upon our Opportunistic Reasoning Platform, and we have presented the main reasoning components that take part in the design support system. In the next section we move into more technical aspects of the implementation of ODEA, paying special attention to the mechanisms that have supported the knowledge-based reasoning processes over the Web.

4 Implementation of ODEA

In this section we enter into more technical matters by describing, first of all the main implementation issues derived from the need to provide a Web-based user interface. Secondly, we explain how we have integrated the Knowledge Base and additional inferencing facilities required for supporting the event designer. Finally, we briefly present how the external systems have been integrated in ODEA in order to support the collaboration with external providers.

4.1 A Web-based System

The Online Design of Events Application is an events design support system that allows event organisers to use their favourite web browser for designing

events. ODEA has been developed in Java and its reasoning machinery has been encapsulated in a web module that provides an HTML user interface. The system has finally been deployed on a J2EE Application Server [15], see Figure 4.

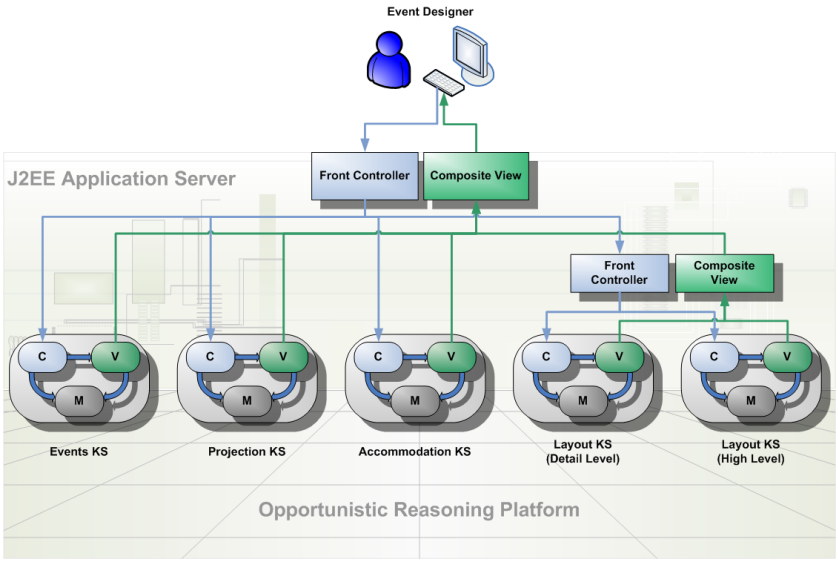

Fig. 4. ODEA deployed on a J2EE application server.

In order to obtain an homogeneous user interface, we have applied the *Composite View* and *Front Controller* design patterns [16], see Figure 4. The Composite View bundles all the Knowledge Source's user interfaces, into a single one. At runtime the different components of the overall interface are generated by the specific Knowledge Sources and are bundled together. The Front Controller design pattern provides the initial point of contact for handling users requests. This Front Controller centralises the actions common to all the views, such as session management or access control and provides a common entry point to every user interaction which is later on delegated to the appropriate Knowledge Source.

Perhaps, the most complex process regarding the encapsulation of the design support system in a web module, has been the generation of HTML views of the data, and the manipulation of the user interaction. In fact, it is often overlooked that HTML is not well suited for building rich user interfaces. Adding dynamic elements requires using other scripting languages, and the development process is tedious and prone to errors. Moreover, the communication between Web servers and browsers, which is based on the Client-Server model, makes it difficult to obtain highly dynamic user interfaces able react upon changes on the Web server.

In ODEA we have limited the impact of these issues by integrating JavaScript code in the HTML user interface. The JavaScript elements provide the means for showing graphics-based calendars, dynamic help information, validating data

types, and even specific parts of the expertise of the Knowledge Sources. The most complete example in this respect is the Events KS. It features, functions to ensure the *start date* is previous to the *end date* of the event; functions that inform the designer about the different types of events as modelled in the Event Types Ontology; functions that ensure the required information has been provided in the appropriate data types; and a function to ensure the selected event type and the number of participants is consistent with our characterisation.

Therefore, despite being based on an HTML user interface, the system provides a dynamic user interface and minimises the Web server overhead through simple scripts executed on the client side. It is worth noting that the inclusion of such a kind of rules (i.e., part of the expertise of the Knowledge Sources) in JavaScript embedded into the HTML user interface, is enabled by the fact that our Opportunistic Reasoning Platform *externalises* the Knowledge Sources which are in charge of their own user interface and can thus apply their expertise to improve it.

4.2 Knowledge Integration in ODEA

ODEA is informed by the set of ontologies we previously introduced. The ontologies have been represented in RDF/RDFS as decided by the consortium of the OBELIX project. In order to manipulate the ontologies as well as to support the persistence and efficient manipulation of the Knowledge Base, ODEA integrates a Sesame server [17] which is deployed on top of a MySQL Database. Sesame provides ODEA with an outstanding support for efficiently storing, querying and reasoning over RDF and RDF Schema.

At runtime, the Knowledge Sources generate RQL queries [18] for retrieving the required information. For instance, the Events KS generates the appropriate query for retrieving the different event types, and the Projection KS takes the user criteria and constructs the corresponding RQL query for retrieving the available projectors that meet the user's criteria from the Knowledge Base.

The instances stored in the Knowledge Base are retrieved at runtime by the Knowledge Sources and are automatically mapped into Java objects. In order to do so, at design time, a set of Java classes were automatically generated based on the ontologies defined. An intermediate module, developed on top of the Sesame API, offers some useful and typically used methods for dealing with the Knowledge Base (e.g., obtain all the instances of a particular concept, obtain the subclasses of a concept, etc.). Additionally, it includes the mechanisms for transforming the instances retrieved in the appropriate Java instances by means of the Java Reflection mechanism [19] using the BeanUtils component developed by the Apache Foundation [20]. As a result, each of the Knowledge Sources that take part in ODEA have the capabilities for manipulating the Knowledge Base efficiently without further complications or any burdening regarding the serialisation format.

Ontologies are an appropriate means for defining static knowledge for they are sufficiently expressive and allow for reusing Knowledge Bases and integrating pre-existing knowledge-based components. However, the ability to define be-

haviour aspects when developing Knowledge-Based Systems is an important, and in many cases essential, requirement [21]:

> "Many intelligent systems are designed primarily to answer queries about large bodies of knowledge. In these cases, ontologies provide most of the representational muscle needed to build complete systems. To build systems that solve real-world tasks, however, we must not only specify our conceptualisations, but also clarify how problem-solving ideally will occur."

Mapping the instances retrieved from the Knowledge Base into Java instances, simplifies to an important extent their manipulation, and additionally enables integrating them into further reasoning processes. In ODEA, each Knowledge Source integrates its own inference engine, Jess [22], and applies its specific inferencing rules to achieve its goals (i.e., support the user in some part of the event design). The Jess engine supports to directly map its workspace to external Java instances. Thus, any change made by a rule execution is directly reflected on the Java instance held on the Blackboard and inversely, any modification made on the Blackboard is automatically reflected in the rule engine's workspace. The rules are applied at runtime over the Java instances obtained from the Knowledge Base and therefore complement the expressivity of ontologies while we maintain their shareability.

4.3 Integration of External Systems

The essentially distributed and modular nature of the Blackboard Model, confers an outstanding support for integrating heterogeneous and remote systems on Blackboard Frameworks. In ODEA we have benefited from such a capability in order to integrate a generic configuration tool and a hotel web service.

We previously introduced the need for solving configuration problems in ODEA. The expertise required to do so is provided by a configuration Problem-Solving Method implementation. The tool comes in two flavours, the first one as a typical software component offering a Java Application Programming Interface (API), and the second one being encapsulated as a Web Service. Both solutions have successfully been tested in our design support system, although in this paper we have uniquely presented the Web Service version for it is more representative for illustrating reasoning over the Web.

The Accommodation KS, does also present an illustrative example on the development of Knowledge-Based Services over the Web. This KS fulfils its role by contacting a (simulated) Hotel Web Service specifically developed for ODEA. At runtime, the design support system, contacts the service in order to check the availability of rooms, and eventually supports booking the desired facilities. This Web Service demonstrates how the underlying Opportunistic Reasoning Platform supports seamlessly integrating external systems in a knowledge-intensive activity over the Web.

The seamless integration of the (Web Services-based) configuration tool and the Hotel Booking Service is ensured by Web Services technologies. To this

end, we have used the Apache Axis component [23]. Thanks to this software component, the generation of WSDL descriptions of the services, as well the creation of clients that use these services was relatively straightforward. In fact, Apache Axis in addition to providing the core software for executing remote Web Services, provides outstanding tools for the automated generation of WSDL descriptions and client skeletons for using the services.

5 Conclusions

In this paper we have described ODEA, a Semantic Web-based events design support system, that supports event organisers in the preparation of meetings, conferences and the like. Event organisation is understood as a designing activity involving diverse actors such as equipments providers, and hotels, in a complex value chain. Informed by a Knowledge Level theory of Design, the $K_L D^E_{V1}$, ODEA supports the event organiser in the smooth exploration of the possible event designs, by guiding the designing process, assisting the user in the specification of problems, solving these problems, contacting external providers and maintaining a coherent state of the overall design.

Throughout the description of our design support system, we have explored the very aspects involved in its development. We have presented the underlying architecture of the system which is based on our Opportunistic Reasoning Platform. We have described how knowledge has been integrated in ODEA, by means of ontologies, represented in RDF/RDFS and stored in a Sesame server, and complemented with inference rules that allow us to overcome their expressivity limitations. We have illustrated how ontologies and Web Services have streamlined the seamless yet effective integration of external agents, notably a remote Problem-Solving Method implementation and a hotel booking service. Finally, we have introduced the issues derived from providing a Web-based user interface, and have presented the approach adopted to overcome them.

In this paper, we have therefore presented an illustrative case study on the development of Knowledge-Based Services for the Web. We have addressed an important number of issues particularly relevant for a variety of fields, including the Semantic Web, eCommerce and AI in Design. Finally, and most importantly, we have shown how the development of Knowledge-Based Services for the Web, able to process semantic information and perform automated reasoning over it, can be performed by applying Semantic Web technologies and reconciling Software and Knowledge Engineering best practices.

References

1. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. Scientific American (5) (2001) 34–43
2. Smithers, T.: Towards a knowledge level theory of design process. In Gero, J.S., Sudweeks, F., eds.: Artificial Intelligence in Design '98, Kluwer Academic Publishers (1998) 3–21

3. Smithers, T.: On knowledge level theories and the knowledge management of designing. In: International Design Conference - Design (2002), Dubrovnik (2002)
4. Newell, A.: The knowledge level. Artificial Intelligence **18**(1) (1982) 87–127
5. Smithers, T.: Synthesis in designing. In Gero, J., ed.: Artificial Intelligence in Design. (2002)
6. Schreiber, A., Wielinga, B.: Configuration design problem solving. IEEE Expert **12**(2) (1997) 49–56
7. Engelmore, R.S., Morgan, A.J.: Blackboard Systems. The Insight Series in Aritificial Intelligence. Addison-Wesley (1988) ISBN: 0-201-17431-6.
8. Aguado, J., Bernaras, A., Smithers, T., Pedrinaci, C., Cendoya, M.: Using ontology research in semantic web applications: a case study. In: Proceedings of the 10th Conference of the Spanish Association for Artificial Intelligence, CAEPIA 2003, and 5th Conference on Technology Transfer, TTIA 2003. (2003)
9. Cendoya, M., Bernaras, A., Smithers, T., Pedrinaci, C., Aguado, J.: Online design of events application trial, version 1. Available from: http://obelix.e3value.com (2003) OBELIX IST-2001-33144 DELIVERABLE 7.7.
10. Altuna, A., Cabrerizo, A., Laresgoiti, I., Peña, N., Sastre, D.: Co-operative and distributed configuration. In: Net.ObjectDays 2004, Erfurt, Germany (2004)
11. Studer, R., Benjamins, R., Fensel, D.: Knowledge engineering: Principles and methods. Data Knowledge Engineering **25**(1-2) (1998) 161–197
12. Gennari, J.H., Tu, S.W., Rothenfluh, T.E., Musen, M.A.: Mapping domains to methods in support of reuse. International Journal of Human-Computer Studies **41**(3) (1994) 399–424
13. Gennari, J.H., Cheng, H., Altman, R.B., Musen, M.A.: Reuse, corba, and knowledge-based systems. International Journal of Human-Computer Studies **49**(4) (1998) 523–546
14. Fensel, D., Benjamins, V.R.: An architecture for reusing problem-solving components. In: European Conference on Artificial Intelligence. (1998) 63–67
15. Johnson, R.: Expert One-on-One J2EE Design and Development. Wiley Publishing (2003)
16. Alur, D., Crupi, J., Malks, D.: Core J2EE Patterns Best Practices and Design Strategies. Sun Microsystems Press (2003)
17. Broekstra, J., Kampman, A., van Harmelen, F.: Sesame: A generic architecture for storing and querying rdf and rdf schema (2002) International Semantic Web Conference (ISWC 2002).
18. Karvounarakis, G., Alexaki, S., Christophides, V., Plexousakis, D., Scholl, M.: Rql: A declarative query language for rdf. In: In The 11th Intl. World Wide Web Conference (WWW2002). (2002)
19. Horstmann, C.S., Cornell, G.: Core Java 2. Volume 2 - Advanced Features. Sun Microsystems Press (2001)
20. The Apache Software Foundation: Jakarta project: Beanutils version 1.4. http://jakarta.apache.org/commons/beanutils/ (2002)
21. Musen, M.A.: Ontologies: Necessary–Indeed Essential–but Not Sufficient. IEEE Intelligent Systems **19**(1) (2004) 77–79
22. Friedman-Hill, E.: Jess in Action. Manning Publications Co. (2003)
23. The Apache Software Foundation: Apache axis version 1.1. http://ws.apache.org/axis/ (2003)