Improving the Usability of Integrated Applications by Using Interactive Visualizations of Linked Data

Heiko Paulheim
SAP Research Darmstadt
Bleichstrasse 8
64283 Darmstadt, Germany
heiko.paulheim@sap.com

ABSTRACT

Today, data is contained in many different, heterogeneous information systems. For turning that data into valuable information, those systems need to be integrated. System integration on the user interface level, e.g., in portals or mashups, allows users to access multiple applications in parallel and explore the information contained therein while reusing familiar and powerful user interfaces. Linked data can significantly add value by allowing uniform access to the data contained in the systems, and by letting reasoners discover hidden, non-explicit knowledge from that data.

In this paper, we present an approach combining both paradigms. A framework for integrating user interfaces is enhanced by the *Semantic Data Explorer*, which is interlinked with the integrated applications in a *hybrid view*. With a quantitative user study conducted in the emergency management area, we show that hybrid visualizations of annotated data help gathering information from integrated systems, even for users who are not trained with ontologies and semantic networks.

Categories and Subject Descriptors

D.2.2 [**Software Engineering**]: Design Tools and Techniques—*User Interfaces*; H.5.2 [**User Interfaces**]: Graphical user interfaces (GUI); I.3.6 [**Methodology and Techniques**]: Interaction Techniques

General Terms

Design, Human Factors

Keywords

Visualization, Linked Data, User Interface Integration, User Experience

1. INTRODUCTION

Today, data is most often contained in different, heterogeneous information systems. A typical information worker has various information needs, which he typically satisfies by looking up information in different of those information systems and combining it manually [31]. Applications are more often isolated from each other than integrated with each other, and they may use different vocabularies and presentation mechanisms for providing information to the end user. Furthermore, the amount of available information is generally very large and growing larger. These factors make information search a task of finding a needle not only in one haystack, but in a stack of somewhat related, heterogeneous haystacks.

Different remedies have been proposed to ease this task. One is the *integration of applications and their existing user interfaces*, e.g., in portals or mash-ups [17]. These approaches have the advantages that a *large degree of reuse* is achieved, as whole applications, including their user interfaces, are reused and combined to new systems. Furthermore, users already trained with the individual applications have a steep learning curve when operating the integrated system. As an additional benefit of UI integration, the reused *powerful user interfaces* do not only provide sophisticated ways for displaying, but also for creating and manipulating data. On the other hand, each of the integrated user interfaces provides access to only a part of the complete information, but there is no single point of interaction with all information.

Another promising approach, developed mainly by the semantic web community, is the *linked data* approach [9]. Information systems can publish their information as linked data – basically RDF representations of the data containing links to data contained in the same or other systems. To this end, the systems provide endpoints which deliver RDF data at dereferencable URIs – i.e., URIs that are used as identifiers for resources and, at the same time, as pointers to additional information on those resources. If formal ontologies are used for defining the categories and relations of linked data, reasoners may run on combinations of linked data providers and draw conclusions from information contained in different systems, i.e., discover *implicit knowledge*.

Applications consuming linked data can use that data, and additional knowledge discovered by reasoning, and provide a meaningful *user interface to the whole set of information* – such as a query interface or a visual browser. However, those interfaces are then isolated from the information systems' original user interfaces. Furthermore, despite the presence of different linked data editors [18, 47], created and edited linked data cannot easily be fed back into the original information systems – thus, in many cases, it is still a one way street from the information system to the end user [7], and the user interfaces to linked data are typically less powerful than original user interfaces. Furthermore, as linked data is an approach for data integration, not to UI integration, the degree of reuse is typically lower, as new user interfaces are required [56].

Table 1 lists some of the main benefits both of integrated UIs and of linked data. It shows that integrated user interfaces could gain

WIMS '11, May 25-27, 2011 Sogndal, Norway

	Integrated User Interfaces	Visualized Linked Data
Powerful legacy UIs	+	-
Large degree of reuse	+	-
Overview on whole dataset	-	+
Inferred implicit knowledge	-	+

Table 1: Comparison of the main benefits that UI integration and linked data can provide.

additional value from the combination with visualized linked data. In this paper, we present the *Semantic Data Explorer*, which visualizes the data contained in integrated applications. The platform underlying the Semantic Data Explorer consumes linked data provided by the applications and uses a reasoner for inferring implicit information. We have integrated the Semantic Data Explorer in a framework for user interface integration. The result is a powerful solution of an integrated information system which provides access to information both through the original user interfaces and through an integrated view on the underlying data.

In a case study from the field of emergency response, where users have to combine large amounts of information from different systems for quickly gathering an overview on a catastrophic situation, we show how visualized linked data and an integrated user interface provide efficient access to information. With a user study, we prove that our approach leads to better usability of integrated information systems, even for users not particularly trained with ontologies and semantic networks.

The rest of this paper is structured as follows. Section 2 provides an overview of related work both on UI integration and on visualization of linked data. Section 3 introduces our framework for user interface integration, as well as the Semantic Data Explorer, and describes their implementation. Section 4 presents a case study from the emergency management domain used for an evaluation with end users, and discusses the evaluation results. We conclude with a summary and an outlook on future work.

2. RELATED WORK

User interface integration as well as visualization of semantic and linked data have been well researched in isolation of each other. This section gives a brief overview of the state of the art in both areas.

2.1 User Interface Integration

Application integration on the user interface level, or UI integration for short, is a technique for integrating software systems in a way that, following Fowler's three layer model [26], all three layers of the integrated software systems (data storage, business logic, and user interface) can be reused [17].

One of the first approaches to UI integration was *Snap Together* [50], an approach which couples the data views of different applications. It uses a set of fixed events and operations to provide various ways of coupling, such as synchronized scrolling and highlighting, coordinated overview and detail views, etc.

Portals are frameworks that provide access to several applications with a common access point, typically combined with features such as single sign on [68]. Applications are encapsulated in *portlets*, e.g., using the JSR-268 standard [33]. Besides the JSR-268 reference implementation *Apache Pluto* [3], examples for well-known portal platforms are *JBoss GateIn Portal* [40], *IBM WebSphere* [37], and *SAP Netweaver Portal* [62].

While portals are targeted at professional developers, *mashup platforms* follow the vision of providing skilled end users with the possibility to assemble different applications [71]. The result is most often web-based, with different views on data running in one web platform and some reasonable amount of interaction possible between them. While the often-cited *Yahoo Pipes* [70] is rather a tool for mashing up data than user interfaces, popular UI mashup platforms are *Google Mashup Editor* [28] (meanwhile discontinued), *JackBe Presto* [39], *Intel MashMaker* [38], and *IBM Mashup Center* [36].

Most of those approaches share the same set of shortcomings. In [17], the authors name the lack of formal models and limited capabilities for event exchange between integrated applications as current shortcomings of UI integration approaches. Therefore, it is hard to achieve seamlessly integrated user interfaces with current approaches, especially if technically heterogeneous applications are involved. In our previous works on UI integration, we have presented an approach using ontologies for overcoming those problems [53, 56]. We will briefly sketch that approach in section 3.1.

2.2 Visualization of Linked Data

Visualization of linked data falls into the category of ontology-based browsing of information, where semantic annotation of data is used for supporting the exploration of that data [57].

As linked data builds upon the RDF standard, visualizing linked data is strongly related to visualizing RDF data. Many ontology authoring tools and suites come with a number of graphical views of RDF and ontology data. For example, the ontology engineering tool *Protï£¡gï£¡*, the most widespread in its field [14], comes with plugins such as *IsaViz*, *OwlViz*, or *Jambalaya*, all providing different visual representations of ontologies and RDF data [44]. Among the variety of other generic RDF visualization tools, two of the best-known are *RDFGravity* [29], and the visualization provided by the W3C's RDF validation service [67]. RDF graphs can also be interactive, as demonstrated in [64].

Linked data is special as it does not consist of a single document holding a whole RDF graph, but is distributed, either across different information systems within an organization, or even across the whole web. Various tools have been developed for accessing linked data which respect this distributed nature of linked data. Examples for straight forward linked data browsers are *Disco* [8] and *Zitgist DataViewer* [51], which essentially provide a navigable triple representation without any graphical visualization. A refined approach is taken by *Sig.Ma* [65], which provides additional features, such as data consolidation and advanced labeling of properties.

Tabulator [6] was one of the first tools for browsing linked data. Its basic view presents a tree-like structure of the starting concept, with each relation unfolding a new level containing the related concepts. Furthermore, it contains specialized map and timeline views for visualizing spatial and temporal relations.

Fenfire [32] provides a straight-forward graph view of linked data, allowing the user to start from a resource and see all the linked resources. The user can follow links to browse the graph of linked data. Fenfire also provides some basic editing functionality of the underlying RDF data.

Humboldt [43] is a tool that allows for querying, searching, and browsing linked data, offering faceted browsing and pivoting (i.e. using one facet of the result as the starting point for the next query) for filtering and exploring results. The authors also present a qualitative evaluation with a group of semantic web researchers, showing that the users can use the tool for finding information in linked data with little effort. *Explorator* [20] is another linked data browser based on facets, which allows users to directly manipulate result sets with set operators. The recently discussed extension *RExplorator* [16] allows for reusing and recombining those facets. A user study presented by the authors shows that the tool is better suitable for users with prior knowledge on the underlying formalisms, such as RDF and ontologies.

visR [69] is a tool for browsing and editing linked data. It features a hybrid view, showing the RDF triples as well as a graph view of the underyling data in parallel in two synchronized views. The authors also present a user study with a group of semantic web engineers where they both evaluate qualitative feedback, as well as quantitative data such as the completion time for different tasks.

DBpedia Mobile [5] is an example for providing linked data based on the user's location. The mobile application provides information from DBpedia [12], a large online database containing information about persons, cities, places, etc., and provides it to the user on a mobile phone. The user may configure different filters according to his information needs.

RelFinder [46] is a specialized tool which, unlike the ones discussed so far, does not use *one*, but *two* concepts within linked data to start with, and finds and visualizes links between those two concepts.

Fusion [19] aims at mapping application models to linked data and thereby providing customized, application specific views on linked data, and also provides means to locally store additional data, which can be merged with data from other endpoints. The user interface provides various means of assistance to the user, such as automatic discovery of paths.

ThinkPedia and *ThinkBase* [35] provide a *hybrid view* on Wikipedia and Freebase, respectively, combining an RDF graph view with the original user interfaces. The authors present a qualitative, but not a quantitative user study. A similar approach is presented with *SemanticWonderCloud* [49], which provides a visual interface to DBPedia, combining explicit relations with statistical information, such as the usage of concepts for tagging.

Bridging information silos with semantic web technologies is a topic also addressed by the *semantic desktop* [63]. In projects such as *Haystack* [59] and *IRIS* [15], tools have been developed which allow for accessing data in different desktop applications by providing explicit semantic annotations and visual representations of those. In [30], an approach is presented which enriches data on the semantic desktop with information drawn from the linked open data cloud.

The approach presented in this paper is different from most of the works discussed above in two main aspects: first, it combines legacy user interfaces and visualized linked data to a *hybrid view* involving arbitrary legacy systems, and second, it is targeted as the domain expert as an end user, not a semantic web expert. Consequently, we present a *quantitative user study* which also encompasses *users who are not experienced in the semantic web area*, and analyze our results with respect to prior experience with ontologies and semantic networks.

3. PROTOTYPE

We have integrated our Semantic Data Explorer into a framework for integration of applications on the user interface level. This framework provides means for combining technologically heterogeneous applications and enabling cross-application interactions while preserving the original interfaces.

3.1 The Integration Framework

Our framework follows the rationale that all parts of existing applications, including their user interfaces, can be reused, and that users can work with mashed-up applications that they already

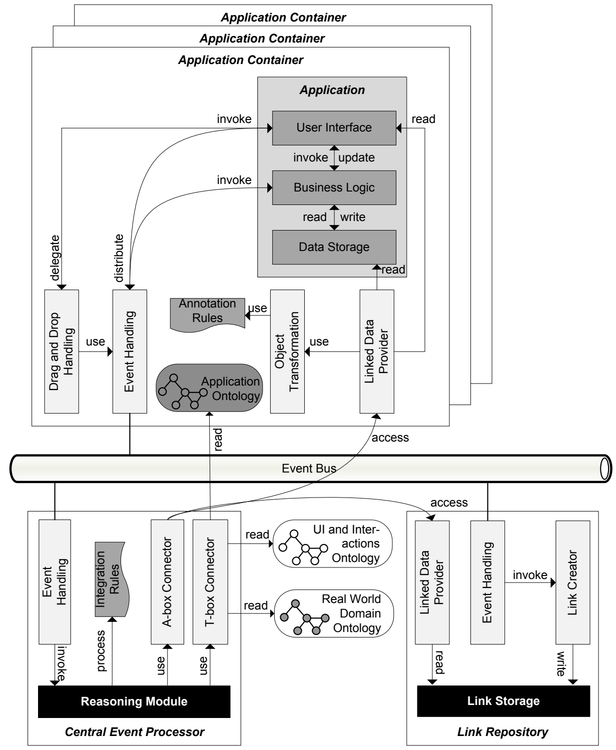


Figure 1: Architecture of our integration framework. The figure shows the containers for Flex and Java applications. Each container may be instantiated multiple times for integrating different applications.

know, instead of having to learn how to operate on a new user interface. While ontologies have been proposed as a means for integrating user interfaces already some years ago [60], our framework is the first running prototype of such an approach. We use ontologies for describing both the technical components of the integrated applications' user interfaces and the information objects they process [56]. The framework is written in Java and based on the OntoBroker infrastructure [21].

Figure 1 shows the framework's basic architecture. Each application which is integrated with our framework runs in a *container*. There are different container implementations for each UI technology, i.e., a Java container, a Flex container, etc. This allows for the integration of heterogeneous UIs to one coherent system [55]. Each container provides different services for performing the application integration:

- An *event handling* service is used to exchange events with other components. To this end, each application is connected to a central event bus. The event handling service is also responsible for semantically annotating each event, so it can be unambiguously interpreted by each recipient. Being the main communication means between applications, events can be raised by and lead to invocations of both an application's user interface and business logic.

- A *drag and drop handling* service provides special functionality for enabling drag and drop across applications, especially across applications developed with different underlying UI technologies, such as Flex and Java [55]. It keeps track of drag and drop operations across applications, makes

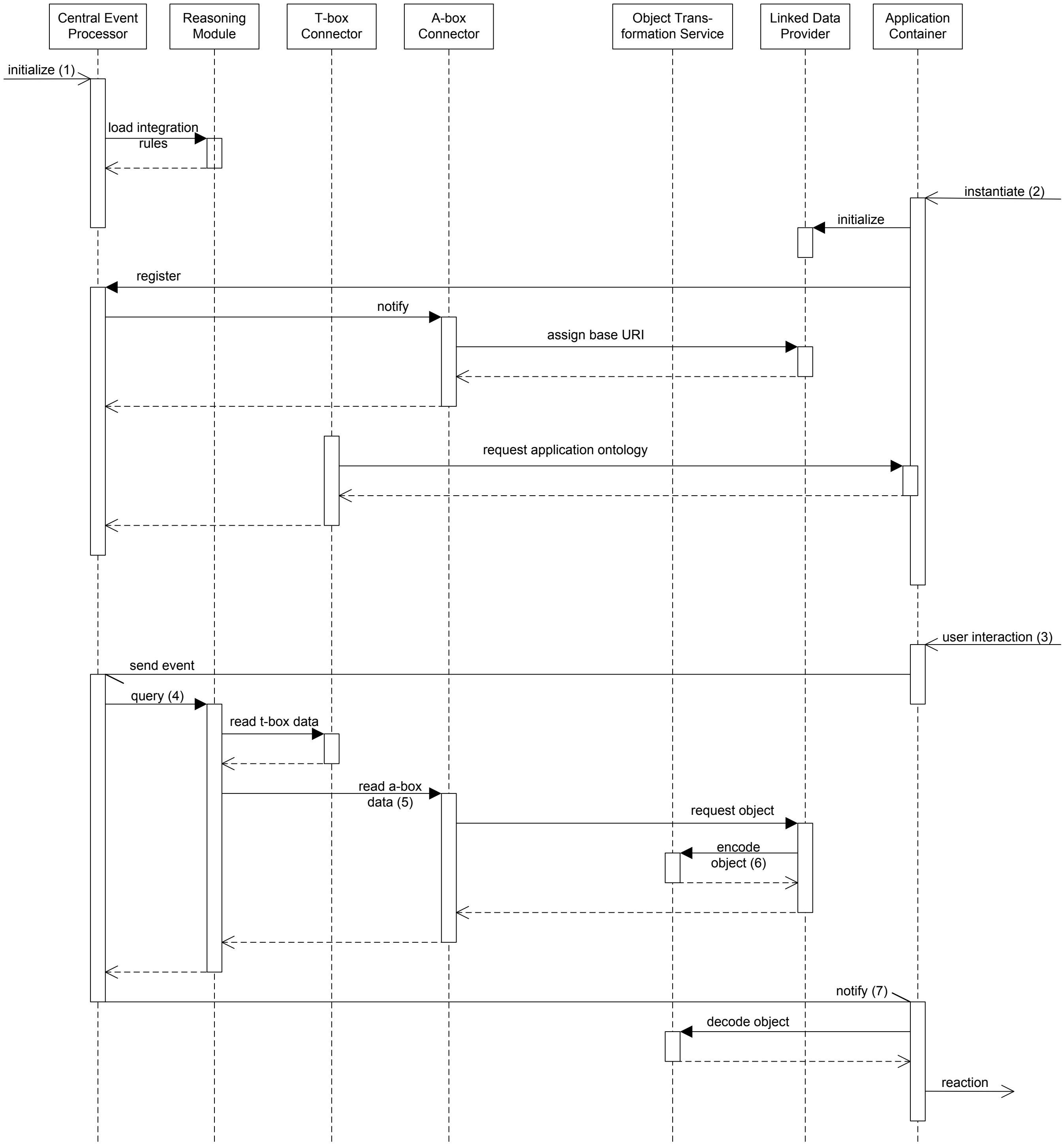


Figure 2: Sequence diagram of the event processing process. As a simplification, it only shows one application container and omits the link repository, as well as the details of the event exchange. Usually, the container sending an event and the container notified to perform a reaction will not be the same.

the respective components provide visual feedback, and handles the objects that are dragged and dropped. The drag and drop service uses the event handling service for coordination with other containers.

- An *object transformation* service transforms information objects from the application into a common RDF format, as well as objects serialized as RDF back into the applications' own information object format. These transformations are defined as a set of annotation rules, which are, in the simplest case, 1:1 mappings of classes and attributes in the applications' class models to categories and relations in the ontology. Annotation rules may also be more complex, e.g. supporting conditional mappings, such as the approach discussed in [34]. This allows for an automated exchange of objects between applications with different underlying object models, as well as different technological platforms [53].

- A *linked data provider* service keeps track of the different UI components, their states, and the information objects they process, and provides that information to the central reasoner as linked data. The linked data endpoint utilizes the object transformation service for creating RDF representations from the underlying applications' objects.

When integrating applications, it is often desirable to establish links between information objects from those applications in a new way that has not been foreseen by the developers of the original applications. For example, one might want to link contact data from an address book to paper authors in a literature database. As neither the address book's nor the literature database's data model provides a means to store those links, they have to be stored in a separate component. To that end, our framework provides a *link repository* where such links may be stored [13]. This repository, like the application containers, provides its data as linked data, and it may exchange events with containers.

One key design rationale of the integration framework was that the integrated applications should be only loosely coupled, i.e., that no direct dependencies should be introduced between applications, e.g., by making applications register event listeners with other applications. Thus, a central event processor has been introduced which serves as an indirection between the integrated applications. By means of the central event processor, no application container communicates directly with other containers, but all communication is coordinated at a central point, thus reducing the complexity and increasing the maintainability of the integrated system [53].

This central event processor uses a reasoner, which computes reactions to the events issued by each application. The reasoner, which is also connected to the central event bus, uses three different types of input for computing the results:

- Different domain ontologies. A domain *ontology of user interfaces and interactions* is used to define basic categories of objects (such as widgets, windows, controls, ...) as well as events (such as select, drag, drop, ...) that can be used to describe user interfaces and their components. A *real world domain ontology* defines the real world items the application deals with (such as books or customers). *Application ontologies* can define concepts for specific applications based on those ontologies.

  Those ontologies are used to annotate both the data provided by the linked data providers and the events emitted by each application. The ontologies are provided to the reasoner by the *T-box connector*.

- Instance data, describing the system's current state, as well as the information objects that are processed by each integrated application. This information is made available by the individual applications' linked data providers (see above) and read into the reasoner by the *A-box connector*. The *A-box connector* addresses the linked data providers dynamically whenever information from any of those endpoints is needed to resolve a query. In our previous work [52], we have shown that this architectural approach scales up to large data sets and allows for fast query answering by reasoners.

- *Integration rules* define the desired interactions between the integrated applications and provide the wiring of those applications. Essentially, an integration rule is an Event-Condition-Action (ECA) rule [2], which uses the vocabulary of the different ontologies for formulating events, conditions, and actions. In our framework, we use F-Logic [1] for formalizing those rules.

The following example of linking views, i.e. highlighting related objects [24], illustrates how the framework works in detail. The underlying integration rule states that upon a selection event with an object, all applications which hold information objects representing the same object are required to highlight those information objects[1]. Figure 2 shows the steps performed to create this interaction:

1. During system startup, the integration rules are loaded into the reasoner.

2. When a new container is instantiated, the corresponding linked data provider is set up and registered at the A-box connector. It is assigned a base URI to use for creating URIs for the objects it provides. Furthermore, the application ontology for the contained application is loaded by the T-box connector.

3. When the user selects an object, the corresponding annotated event is sent via the event bus. The event contains the URIs of the selected information objects and the component that the event was performed with.

4. The central event processor's event exchange service reads the event and queries the reasoner for reactions.

5. The reasoner uses the A-box connector to resolve the URIs of components and information objects delivered with the event and retrieve the corresponding instance data. As the interaction rules may require more information about other components and related information objects, the A-box connector queries the applications' linked data providers for the required data, while the required T-box data can be delivered directly by the T-box connector.

6. When an object is requested from a linked data provider, the container's object transformation services are used to provide the required information, i.e. an RDF representation of the requested object.

7. Based on the results of the integration rules, the reasoner answers the query. In the particular example, the result list contains the action of highlighting objects related to the object selected by the user.

[1]A detailed example on the formal definition of such an integration rule can be found in [52].

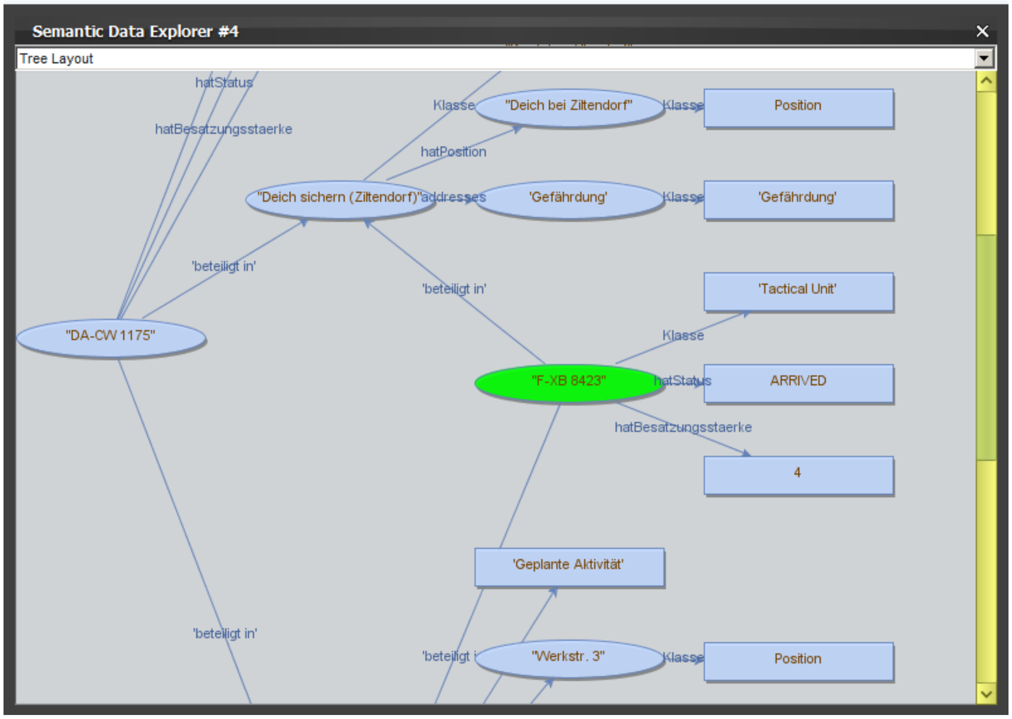


Figure 3: Screen shot of the Semantic Data Explorer (SDE)

8. The central event processor sends events to notify the applications to perform the respective actions.

9. Each notified application receives the event, decodes the object contained therein using its object transformation service, and performs the highlighting action as a reaction.

Other interactions between applications can be defined and processed similarly. Those include interactions involving data input actions requiring input in different applications (e.g. customer data in one application and accounting data in a different application) or drag and drop between applications. For the latter case, our framework also allows highlighting suitable drop locations and augmenting them with tool tips when the user starts dragging an object, thus supporting the user in understanding the interaction possibilities of the integrated system [52].

Creating links in the integrated system works similar to cross-application interactions: an integration rule is defined, stating that, e.g., whenever the user drops an address book entry in the address book application to an author in a bibliography application, a link between the two should be created. When the drop event occurs, the reasoner evaluates the rule and sends the respective event to the link repository, which creates and stores the link and makes it available through its linked data provider service.

3.2 The Semantic Data Explorer (SDE)

The Semantic Data Explorer visualizes information objects, their attributes and relations to other objects, as a semantic network. It uses the RDF visualization proposed by the W3C [66], showing objects (or resources) as ellipses and data values in rectangles, connected by directed edges. Instead of URIs, labels (i.e., `rdfs:labels`) are used as captions for resources.

As the SDE is an exploration and explanation tool merely for domain experts, rather than a debugging device for developers, it uses the concepts and labels defined in a domain ontology, not the class and attribute names of the underlying objects. It thus reflects a *conceptual* view on the information, not a *technical* one.

The Semantic Data Explorer has been developed an extension to the integration framework shown above. From the framework's point of view, it is implemented as a special container, which uses a direct interface to the reasoner for reasons of performance (unlike the other containers, which do not directly access the reasoner). Figure 4 shows how the semantic data explorer is integrated in the framework. A *data access* component is used as an interface to the reasoner. It accesses the data from both the applications as well as the link repository indirectly through the reasoner. The same holds for the T-box data, which is especially used for labeling nodes and edges in the semantic network. The SDE is also connected to the event exchange bus to allow interactions with the other applications:

1. When the user drags and drops an object from an application onto the SDE, a graph view of that object is shown by the SDE, containing the object itself and its directly connected neighbors. To get a deeper view of the object's connections, the user can double click nodes in the graph to expand and collapse them.

2. When the user selects an object in an application, and that object is contained in the graph currently shown by the SDE, the corresponding node in the graph is highlighted. Likewise, if the user selects a node in the graph, the corresponding objects in other applications are highlighted. This allows the user to keep track of information objects in the conceptual view provided by the SDE, as well as in the original applications' user interfaces.

These interactions are implemented using the same mechanisms as for interactions between two applications in our integration frame-

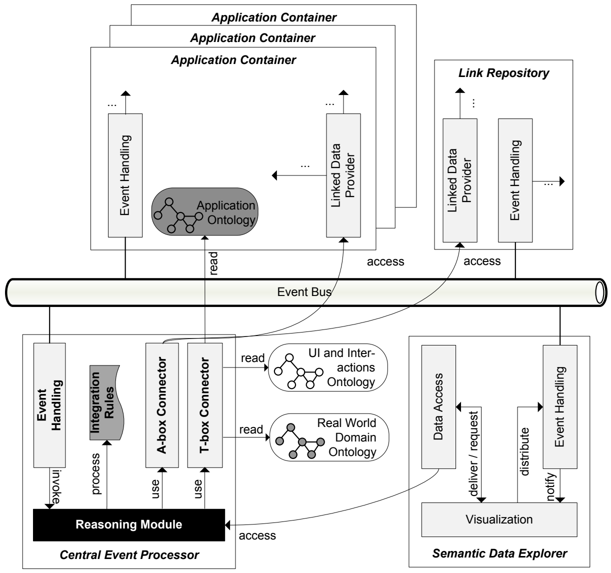
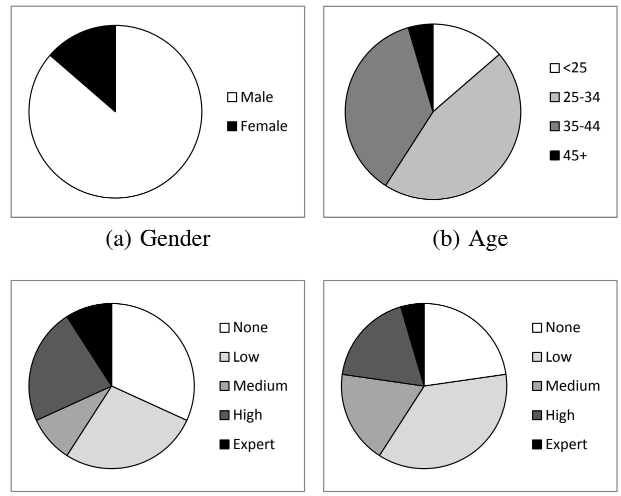


Figure 4: Architecture of the Semantic Data Explorer (bottom right) and its integration into the framework. The details of application containers and the link repository are omitted; they are depicted in figure 1.



(a) Gender (b) Age

(c) Experience with Ontologies (d) Experience with Semantic Networks

Figure 5: Participants of the user study by age, gender, and self-assessment of prior knowledge about ontologies and semantic networks.

work. Therefore, no particular adaptations have to be made to integrated applications for adding the semantic data explorer; the SDE is thus transparent for the applications.

Technically, displaying as well as expanding a node in the semantic data explorer results in a call on the reasoner, querying for outgoing and incoming object relations as well as for data relations. Unlike querying the individual linked data endpoints, interfacing the reasoner has the advantages that implicit, inferred relations and class information are returned as well. From a software engineering point of view, the reasoner again serves as an indirection between the SDE and the applications holding the data to visualize.

4. EVALUATION

To show that the Semantic Data Explorer provides a valuable extension to integrated information systems, we have conducted a user study for which we added the SDE to the integrated emergency management system *SoKNOS*[2] [13, 23, 54]. That system was built as a prototype in a research project conducted with several software development companies, universities and other research institutions, as well as experts from the emergency management domain, such as firefighters and police officers.

4.1 Scenario

The SoKNOS prototype is based on the framework introduced in section 3.1 and consists of 20 different integrated applications. It shows integrated information on emergency situations in maps, tables, and timelines, where external information sources, such as web services or data gathered by web crawlers, can also be integrated [27]. Furthermore, simulations (e.g., of the spreading of fires or floodings) can be executed, messages from and to the operational resources (such as fire brigade cars or helicopters) in the field can be handled, and information can be shared across different organizations involved in larger disasters.

For the evaluation, we used two particular applications in SoKNOS: a *mission account* application which is used for viewing problems and measures, as well as assigned tactical units (such as fire brigade cars), and a resource management application used for browsing available tactical units, both from the operator's own organization as well as from other supporting organizations. Figure 6 shows the two applications. Assignments of units and measures can be made by dragging and dropping a unit from the resource management application and dropping it onto a measure in the mission account application. Technically, this creates a link between the data managed by the two applications. The link can be removed by dragging and dropping an assigned unit back to the resource management application.

The two applications used in the evaluation are developed on different technological platforms (Flex and Java). Details on the coupling of those heterogeneous applications can be found in [55].

The scenario of the evaluation deals with the assignment of operational resources to measures, e.g. sending helicopters to evacuate people from a roof top, or assigning fire brigade cars to individual areas of fire. Planning such an assignment is an important task in emergency management – an optimal assignment is needed for avoiding overload of units as well unnecessary idle times. To this end, the person doing the planning needs a good overview on the current assignments.

4.2 Evaluation Setup

In the course of the SoKNOS project, we have conducted interviews with domain experts who deal with emergency management systems. From those interviews, the requirements for the prototype have been derived. The tasks the users had to perform in our evaluation are also based on those interviews, i.e., they are realistic tasks that the end users need support for. From the field of managing assignments of tactical units to measures, the following three types of tasks have been chosen for the evaluation:

Type 1 Find out how many units are assigned to measure M.

[2] *Service-orientierte ArchiteKturen zur Unterstützung von Netzwerken im Rahmen Oeffentlicher Sicherheit*, German for *service oriented architectures for supporting networks in the area of public security*.

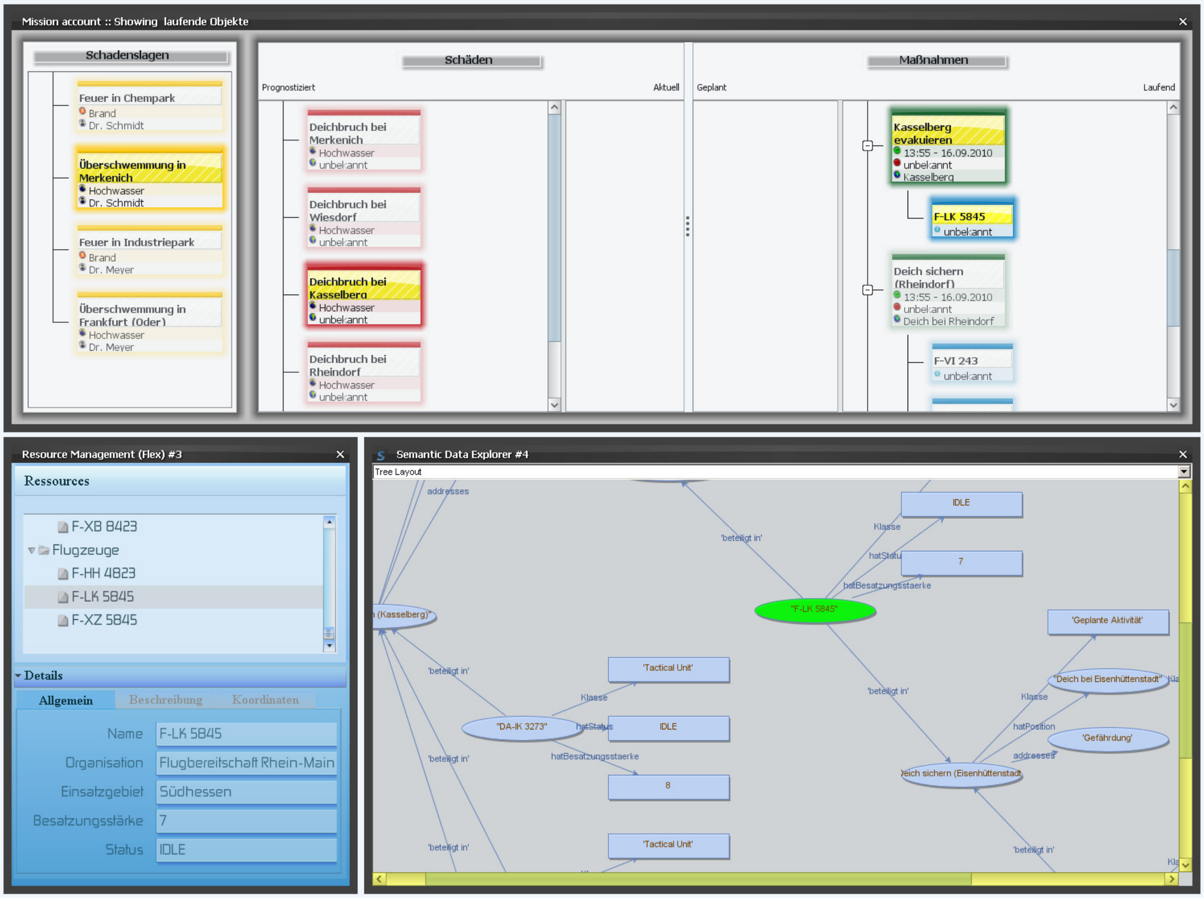


Figure 6: Screenshot of the evaluation setup. The setup consists of two applications – a mission account application (top) and a ressource application (bottom left) as well as the Semantic Data Explorer (bottom right). Items are parallely highlighted in different applications. The SDE has been presented to the users for only half of the tasks.

Type 2 Find out how many measures a unit U is assigned to.

Type 3 Find out how many of the units assigned to measure M are also assigned to at least one other measure.

The answer for tasks of type 1 can be obtained by looking it up in the mission account application only. The answer for tasks of type 2 can be solved with one interaction between the resource management application and the mission account application (e.g., by first selecting the tactical unit and then counting the number of highlighted related measures). The answer for tasks of type 3 requires multiple interactions between the two applications (e.g., first finding the relevant tactical units in the mission account application, and then repeating task 2 for each of those units).

Our hypotheses were that at least the more complex tasks of type 2 and 3 are performed faster with the use of the SDE, that the error rate is lower for tasks of type 2 and 3 when using the SDE, and that the use of the SDE increases the users' satisfaction.

The evaluation was performed with 22 participants between 22 and 52 years, 3 of which were female and 19 of which were male. When asked to self-assess their experience with ontologies and semantic networks, the majority of the participants ranked their experience below average (see figure 5).

To test our hypotheses, each user had to perform two blocks of six tasks, each block consisting of two tasks of each type. One block had to be performed using the SDE, the other without using the SDE. The test persons were split into two groups: One group started with the block of tasks using the SDE, the other with the block of tasks without using the SDE. The assignment of the individual tasks to the blocks was varied as well. When using the SDE, the users were advised to use it for all tasks, even if they felt they could perform those tasks without the SDE as well (and maybe even better). Furthermore, users were offered to use pen and paper if needed.

At the beginning of the evaluation, users were shown how to operate the system, in a way that an instructor showed them how to find the kind of information that was asked for later on the tasks. The group that started with the SDE was presented the whole system including the SDE at once, while the other group was presented the system without the SDE first, and got the introduction to the SDE at the beginning of the second block of tasks.

After each block of six tasks, the users were handed out a questionnaire for evaluating their user experience, using the questionnaire described in [45]. It consists of 20 questions which are used to quantify user experience according to five factors: attractiveness, perspicuity, efficiency, stimulation, and novelty.

When the users performed the tasks, the system recorded the times needed to perform the task and the number of errors. The instructor additionally recorded for which tasks the users made use of the offered pen and paper to take notes. For each user, the evalu-

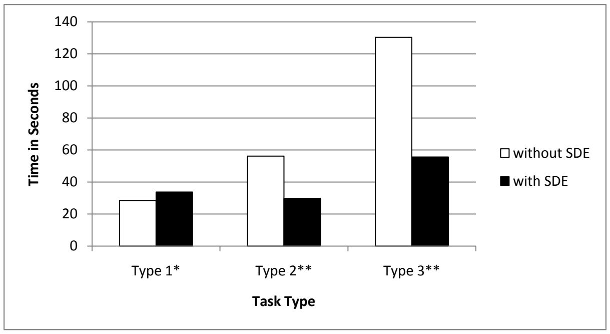


Figure 7: Average task completion times in seconds (*: p<0.05, **: p<0.01).

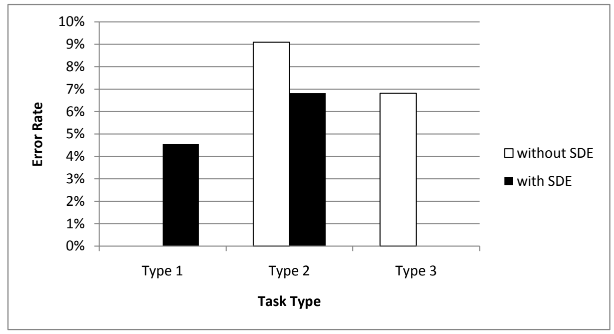


Figure 8: Average error rates. None of the results are statistically significant.

ation, including filling out the questionnaires, took between 20 and 30 minutes.

4.3 Evaluation Results

Figure 7 shows the average task completion times for the different types of tasks, each with and without SDE. It can be observed that the task completion time for tasks of type 2 and 3 are highly significantly reduced ($p<0.01$ using a two-tailed t-test). Especially for the type 3 tasks, the task completion time is reduced by more than 50% when using the SDE. For type 1 tasks, the non-SDE variant is faster ($p<0.05$).

For solving the tasks of type 3 without the SDE, 17 out of 22 participants made use of pen and paper for taking down intermediate results, as opposed to none of the participants using the SDE. This shows that those tasks are particular difficult to solve, and that there is a need for additional support which is provided by the SDE. For the other tasks, pen and paper was used by almost none of the participants.

Figure 8 shows the error rates (i.e., the percentage of tasks of each type that have been solved incorrectly). For type 1 tasks, there are more errors with the SDE, while the error rate is lower for type 2 and 3 tasks. However, none of those results are statistically significant using a χ^2 test.

Figure 9 shows the results from the user experience questionnaires [45]. We have compared the results of both times the users filled the questionnaire, once with and once without the SDE. It can be observed that the users rated the application with the SDE significantly more attractive, perspicuous, efficient, stimulating, and novel (all $p<0.01$ using a two-tailed t-test).

To examine how much the Semantic Data Explorer helps users that are not familiar with ontologies and semantic web, we created a subset of our evaluation results containing only those 11 users that assessed their knowledge on ontologies *and* semantic networks less than medium. For that group of users, the reduction of task completion time for type 2 and type 3 tasks was still significant ($p<0.01$). The evaluation of the user experience questionnaires proved improvements in attractiveness, efficiency, and novelty (all $p<0.01$), while the improvements in perspicuity and stimulation could still be observed, but only on a non-statistically significant level. Neither the task completion times nor the questionnaire results of experts and non-experts differed significantly.

In summary, our original hypotheses on task completion times and on user satisfaction were supported by the evaluation. Our hypotheses on task error rates was not supported. A possible explanation is that the overall number of errors is very low, both with and without the SDE – less than 10% of all tasks were solved incorrectly in total. The participants were neither asked to perform the tasks as quickly nor as correct as possible, and most of them obviously favored correct answers to quick task completion and took their time to answer the questions very thoroughly, which leads to a significant difference in task completion time, as discussed above, at a constant error rate.

The participants were asked for additional comments on possible improvements of the Semantic Data Explorer. One of the most often mentioned criteria was to use meaningful colors and symbols in the SDE – in the prototype, only the currently active node is highlighted in color, and all nodes, regardless of the type of object they represent, are drawn as ellipses, and data values are shown as rectangles. Other users opted for better labels to edges (which at the moment are marked with the `rdfs:label` of the corresponding relation), and a better arrangement of objects when the graph grows larger.

5. CONCLUSION AND OUTLOOK

In this paper, we have introduced the vision of improving integrated user interfaces with graphical visualizations of linked data. The prototype shown in this paper uses a *hybrid view* that combines legacy applications' user interfaces with the Semantic Data Explorer (SDE), which shows a semantic network view on the underlying data. With this prototype, users can browse the information contained in integrated information systems using those systems' original user interfaces and the graphical visualization in parallel, where the views are coordinated so that related information is highlighted in all views. As a reasoner is used to create the semantic network view, the graphical visualization can make implicit information contained in the applications explicit and show non-obvious relations to the user.

We have evaluated our approach using a scenario from the emergency management domain with a group of users, many of which assessed themselves to have no or only little knowledge about ontologies and semantic networks. Our evaluation has shown that using semantic networks for browsing information can lead to substantial improvements both in task completion time and in the users' satisfaction. Other than user studies with comparable tools [20, 43, 69], we have involved non-expert users (w.r.t. RDF and ontologies) and shown that the observed improvements are still significant for those users. Unlike claimed by, e.g., schraefel and Karger [48], we have shown that a simple graph-based visualization of RDF does provide additional value for users.

As the evaluation suggests, using linked data and hybrid visualization can add extra value to existing software applications with comparatively little effort. As discussed by [50], the interface an application has to implement to be used in such a setting is not too

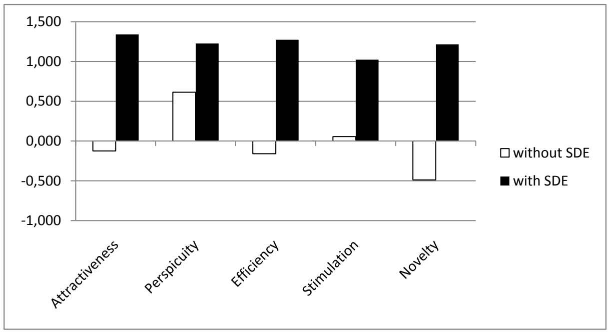


Figure 9: Results of the questionnaires on user experience. All differences are statistically significant with p<0.01. The value space for each criterion ranges from -3 to $+3$.

large, and linked data endpoints may also be added to existing applications with little efforts [4]. Therefore, the approach presented in this paper is a valid approach for building integrated information systems with valuable support for information workers. Compared to developing a whole new user interface, the development efforts of integrating existing user interfaces and combining them with a graph-based view of the underlying linked data, is rather small.

The prototype developed for the evaluation uses desktop applications, but the concept can be carried over to web applications as well when coupling the semantic data explorer with a portal or mashup farmework. It may also be beneficial to enrich linked data provided by information systems internal to an organization (which is linked *closed* data) with additional information from linked *open* data. Furthermore, with the use of ontologies providing multilingual labels, the semantic data explorer can be internationalized with little additional effort. Mappings to other ontologies could help providing personalized visualizations using different vocabularies and conceptualizations.

At the moment, links between information objects in the integrated applications are either created explicitly by the user, e.g. by dragging one object to the other for creating the link, or by the developer ensuring that information objects referring to the same real world object, such as a customer, have the same URI in the different information systems. A more versatile approach could run an instance matcher [25] on the different data sources upfront to create useful links.

Although the evaluation in this paper has shown that significant improvements regarding task completion time and user satisfaction can be achieved with our approach, we could not validate our hypothesis that the error rate could also be reduced by using the Semantic Data Explorer. As the users were in a non-stressful situation, they solved each task carefully, so the overall error rate both with and without the SDE was rather low. We believe that conducting a second user study including stress factors, such as a fixed time limit for each task, would provide additional valuable insights.

Among the informal user feedback gathered, the suggestion to use color coding and symbols for distinguishing different types of objects in the semantic network was one of the improvements which has been asked for most often. As the current prototype only uses a very basic visualization which makes hardly any use of different symbols and colors, more sophisticated and domain-specific visualizations are possible. As the literature suggests many different visualization strategies for semantic networks [42], we aim at developing and evaluating different visualizations. Another request articulated by the participants aimed at more understandable labels for the edges. Using methods from ontology verbalization [41] could provide improvements at that point.

The performance of our semantic data explorer has been experienced sufficiently fast by our test persons; in fact, it takes less than a second to expand or display a node. However, for even more fluent interactions and for more anticipatory layouting of the resulting graph, look-ahead mechanisms could be employed.

In summary, we believe that integrated user interfaces and linked data are a perfect match to build future information systems. The evaluation results discussed in this paper show that this combination is a valuable strategy even for end users who have no particular experience in the semantic web area.

Acknowledgments

The work presented in this paper has been partly funded by the German Federal Ministry of Education and Research under grant no. 01ISO7009 and 01IA08006. The author would like to thank Lars Meyer for his support in implementing the Semantic Data Explorer prototype and conducting the user study, Bettina Laugwitz for her valuable advice on planning the user study and for patiently answering tons of questions on statistics, and Sebastian Döweling, Andreas Faatz, and Florian Probst for their valuable remarks on different stages of this paper.

6. REFERENCES

[1] J. Angele and G. Lausen. Ontologies in F-Logic. In S. Staab and R. Studer, editors, *Handbook on Ontologies*, International Handbooks on Information Systems, chapter 3, pages 45–70. Springer, 2nd edition edition, 2009.

[2] D. Anicic and N. Stojanovic. Towards Creation of Logical Framework for Event-Driven Information Systems. In J. Cordeiro and J. Filipe, editors, *ICEIS 2008 - Proceedings of the Tenth International Conference on Enterprise Information Systems, Volume ISAS-2, Barcelona, Spain, June 12-16, 2008*, pages 394–401, 2008.

[3] Apache Software Foundation. Welcome to Pluto. `http://portals.apache.org/pluto/`, 2010. Accessed July 19th, 2010.

[4] S. Auer, S. Dietzold, J. Lehmann, S. Hellmann, and D. Aumueller. Triplify: Light-Weight Linked Data Publication from Relational Databases. In *WWW '09: Proceedings of the 18th international conference on World wide web*, pages 621–630, New York, NY, USA, 2009. ACM.

[5] C. Becker and C. Bizer. DBpedia Mobile - A Location-Aware Semantic Web Client. In *Proceedings of the Semantic Web Challenge at ISWC 2008*, 2008.

[6] T. Berners-Lee, Y. Chen, L. Chilton, D. Connolly, R. Dhanaraj, J. Hollenbach, A. Lerer, and D. Sheets. Tabulator: Exploring and Analyzing linked data on the Semantic Web. In Rutledge et al. [61].

[7] T. Berners-Lee, R. Cyganiak, M. Hausenblas, J. Presbrey, O. Seneviratne, , and O.-E. Ureche. Realising A Read-Write Web of Data. `http://web.mit.edu/presbrey/Public/rw-wod.pdf`, 2009. Accessed September 30th, 2010.

[8] C. Bizer and T. GauSS. Disco - Hyperdata Browser. `http://www4.wiwiss.fu-berlin.de/bizer/ng4j/disco/`, 2007. Accessed September 30th, 2010.

[9] C. Bizer, T. Heath, and T. Berners-Lee. Linked Data - The Story So Far. *International Journal on Semantic Web and Information Systems*, 5(3):1–22, 2009.

[10] C. Bizer, T. Heath, T. Berners-Lee, and K. Idehen, editors. *Proceedings of the WWW2009 Workshop on Linked Data on the Web*, volume 538 of *CEUR-WS*, 2009.

[11] C. Bizer, T. Heath, K. Idehen, and T. Berners-Lee, editors. *Proceedings of the WWW2008 Workshop on Linked Data on the Web*, volume 369 of *CEUR-WS*, 2008.

[12] C. Bizer, J. Lehmann, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, and S. Hellmann. DBpedia - A Crystallization Point for the Web of Data. *Journal of Web Semantics*, 7(3):154–165, 2009.

[13] S. Braune, A. D. Brucker, G. Kleser, K. Li, T. D. Meijler, H. Paulheim, and F. Probst. A Service-oriented Architecture for Emergency Management Systems. In *Software Engineering 2011 - Workshops*, 2011. To appear.

[14] J. Cardoso. The Semantic Web Vision: Where Are We? *IEEE Intelligent Systems*, 22(5):84–88, 2007.

[15] A. Cheyer, J. Park, and R. Giuli. IRIS: Integrate. Relate. Infer. Share. In Decker et al. [22].

[16] M. Cohen and D. Schwabe. RExplorator - supporting reusable explorations of Semantic Web Linked Data. In Polleres and Chen [58].

[17] F. Daniel, J. Yu, B. Benatallah, F. Casati, M. Matera, and R. Saint-Paul. Understanding UI Integration: A Survey of Problems, Technologies, and Opportunities. *IEEE Internet Computing*, 11(3):59–66, 2007.

[18] S. Davies, J. Hatfield, C. Donaher, and J. Zeitz. User Interface Design Considerations for Linked Data Authoring Environments. In C. Bizer, T. Heath, T. Berners-Lee, and M. Hausenblas, editors, *Proceedings of the WWW2010 Workshop on Linked Data on the Web*, volume 628 of *CEUR-WS*, 2010.

[19] S. F. C. de Araújo, G.-J. Houben, D. Schwabe, and J. Hidders. Fusion – Visually Exploring and Eliciting Relationships in Linked Data. In P. F. Patel-Schneider, Y. Pan, P. Hitzler, P. Mika, L. Zhang, J. Z. Pan, I. Horrocks, and B. Glimm, editors, *ISWC 2010*, volume 6496 of *LNCS*. Springer, 2010.

[20] S. F. C. de Araújo and D. Schwabe. Explorator: a tool for exploring RDF data through direct manipulation. In Bizer et al. [10].

[21] S. Decker, M. Erdmann, D. Fensel, and R. Studer. Ontobroker: Ontology Based Access to Distributed and Semi-Structured Information. In R. Meersman, Z. Tari, and S. M. Stevens, editors, *Database Semantics - Semantic Issues in Multimedia Systems, IFIP TC2/WG2.6 Eighth Working Conference on Database Semantics (DS-8), Rotorua, New Zealand, January 4-8, 1999*, volume 138 of *IFIP Conference Proceedings*, pages 351–369. Kluwer, 1999.

[22] S. Decker, J. Park, D. Quan, and L. Sauermann, editors. *Proceedings of the ISWC 2005 Workshop on The Semantic Desktop - Next Generation Information Management & Collaboration Infrastructure*, volume 175 of *CEUR-WS*, 2005.

[23] S. Döweling, F. Probst, T. Ziegert, and K. Manske. SoKNOS - An Interactive Visual Emergency Management Framework. In R. D. Amicis, R. Stojanovic, and G. Conti, editors, *GeoSpatial Visual Analytics*, NATO Science for Peace and Security Series C: Environmental Security, pages 251–262. Springer, 2009.

[24] S. G. Eick and G. J. Wills. High Interaction Graphics. *European Journal of Operational Research*, 84:445–459, 1995.

[25] A. Ferrara, D. Lorusso, S. Montanelli, and G. Varese. Towards a Benchmark for Instance Matching. In P. Shvaiko, J. Euzénat, F. Giunchiglia, and H. Stuckenschmidt, editors, *Proceedings of the 3rd International Workshop on Ontology Matching (OM-2008)*, volume 431 of *CEUR-WS*, pages 37–48, 2008.

[26] M. Fowler. *Patterns of Enterprise Application Architecture*. Addison Wesley, Boston, MA, 2003.

[27] C. Fritz, C. Kirschner, D. Reker, A. Wisplinghoff, H. Paulheim, and F. Probst. Geospatial Web Mining for Emergency Management. In *GIScience 2010 - Extended Abstracts*, 2010.

[28] Google Inc. Google Mashup Editor. `http://code.google.com/gme/`, 2010. Accessed July 20th, 2010.

[29] S. Goyal and R. Westenthaler. RDF Gravity (RDF Graph Visualization Tool). `http://semweb.salzburgresearch.at/apps/rdf-gravity/`, 2004. Accessed September 27th, 2010.

[30] T. Groza, L. Drăgan, S. Handschuh, and S. Decker. Bridging the Gap between Linked Data and the Semantic Desktop. In A. Bernstein, D. R. Karger, T. Heath, L. Feigenbaum, D. Maynard, E. Motta, and K. Thirunarayan, editors, *The Semantic Web - ISWC 2009*, volume 5823 of *LNCS*, pages 827–842. Springer, 2009.

[31] L. Hardman, J. van Ossenbruggen, R. Troncy, A. Amin, and M. Hildebrand. Interactive Information Access on the Web of Data. In *Proceedings of the WebSci'09: Society On-Line*, 2009.

[32] T. Hastrup, R. Cyganiak, and U. Bojars. Browsing Linked Data with Fenfire. In Bizer et al. [11].

[33] S. Heppner. JSR 286: Portlet Specification 2.0. `http://www.jcp.org/en/jsr/detail?id=286`, 2008. Accessed July 19th, 2010.

[34] G. Hillairet, F. Bertrand, and J. Y. Lafaye. Bridging EMF applications and RDF data sources. In E. F. Kendall, J. Z. Pan, M. Sabbouh, L. Stojanovic, and K. Bontcheva, editors, *Proceedings of the 4th International Workshop on Semantic Web Enabled Software Engineering (SWESE)*, 2008.

[35] C. Hirsch, J. Hosking, and J. Grundy. Interactive Visualization Tools for Exploring the Semantic Graph of Large Knowledge Spaces. In S. Handschuh, T. Heath, and V. Thai, editors, *Workshop on Visual Interfaces to the Social and the Semantic Web (VISSW2009)*, volume 443 of *CEUR-WS*, 2009.

[36] IBM Corporation. IBM Enterprise Mashups - IBM Mashup Center. `http://www.ibm.com/software/info/mashup-center/`, 2010. Accessed July 21st, 2010.

[37] IBM Corporation. IBM Web Portal software from WebSphere. `http://www.ibm.com/websphere/portal`, 2010. Accessed July 20th, 2010.

[38] Intel Corporation. Intel Mash Maker. `http://mashmaker.intel.com/`, 2010. Accessed July 21st, 2010.

[39] JackBe Corporation. JackBe Presto. `http://www.jackbe.com/products/`, 2009. Accessed July 21st, 2010.

[40] JBoss Community. GateIn Portal. `http://www.jboss.org/gatein/`, 2010. Accessed July 20th, 2010.

[41] K. Kaljurand and N. E. Fuchs. Verbalizing OWL in Attempto Controlled English. In C. Golbreich, A. Kalyanpur, and B. Parsia, editors, *Proceedings of the OWLED 2007 Workshop on OWL: Experiences and Directions, Innsbruck, Austria, June 6-7, 2007*, volume 258 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2007.

[42] A. Katifori, C. Halatsis, G. Lepouras, C. Vassilakis, and E. G. Giannopoulou. Ontology Visualization Methods - A Survey. *ACM Comput. Surv.*, 39(4), 2007.

[43] G. Kobilarov and I. Dickinson. Humboldt: Exploring Linked Data. In Bizer et al. [11].

[44] M. Lanzenberger, J. Sampson, and M. Rester. Visualization in Ontology Tools. In *Complex, Intelligent and Software Intensive Systems, International Conference*, pages 705–711, Los Alamitos, CA, USA, 2009. IEEE Computer Society.

[45] B. Laugwitz, T. Held, and M. Schrepp. Construction and Evaluation of a User Experience Questionnaire. In A. Holzinger, editor, *HCI and Usability for Education and Work, 4th Symposium of the Workgroup Human-Computer Interaction and Usability Engineering of the Austrian Computer Society (USAB 2008)*, volume 5298 of *Lecture Notes in Computer Science*, pages 63–76. Springer, 2008.

[46] S. Lohmann, P. Heim, T. Stegemann, and J. Ziegler. The RelFinder User Interface: Interactive Exploration of Relationships between Objects of Interest. In *IUI '10: Proceeding of the 14th international conference on Intelligent user interfaces*, pages 421–422, New York, NY, USA, 2010. ACM.

[47] M. Luczak-Rösch and R. Heese. Linked Data Authoring for NonExperts. In Bizer et al. [10].

[48] mc schraefel and D. Karger. The Pathetic Fallacy of RDF. In Rutledge et al. [61].

[49] R. Mirizzi, A. Ragone, T. D. Noia, and E. D. Sciascio. Semantic Wonder Cloud: Exploratory Search in DBpedia. In F. Daniel and F. M. Facca, editors, *Current Trends in Web Engineering - 10th International Conference on Web Engineering ICWE 2010 Workshops*, volume 6385 of *LNCS*, pages 138–149. Springer, 2010.

[50] C. North and B. Shneiderman. Snap-together visualization: a user interface for coordinating visualizations via relational schemata. In *AVI '00: Proceedings of the working conference on Advanced visual interfaces*, pages 128–135, New York, NY, USA, 2000. ACM.

[51] Open Linked Software. Zitgist DataViewer. `http://zitgist.com/products/dataviewer/dataviewer.html`, 2009. Accessed September 30th, 2010.

[52] H. Paulheim. Efficient Semantic Event Processing: Lessons Learned in User Interface Integration. In L. Aroyo, G. Antoniou, E. Hyvönen, A. ten Teije, H. Stuckenschmidt, L. Cabral, and T. Tudorache, editors, *The Semantic Web: Research and Applications (ESWC 2010), Part II*, volume 6089 of *LNCS*, pages 60–74. Springer, 2010.

[53] H. Paulheim. Seamlessly Integrated, but Loosely Coupled - Building UIs from Heterogeneous Components. In *ASE '10: Proceedings of the IEEE/ACM International Conference on Automated Software Engineering*, pages 123–126, New York, NY, USA, 2010. ACM.

[54] H. Paulheim, S. Döweling, K. Tso-Sutter, F. Probst, and T. Ziegert. Improving Usability of Integrated Emergency Response Systems: The SoKNOS Approach. In *Proceedings "39. Jahrestagung der Gesellschaft für Informatik e.V. (GI) - Informatik 2009"*, volume 154 of *LNI*, pages 1435–1449, 2009.

[55] H. Paulheim and A. Erdogan. Seamless Integration of Heterogeneous UI Components. In *Proceedings of the 2nd ACM SIGCHI Symposium on Engineering Interactive Computing Systems (EICS 2010)*, pages 303–308. ACM, 2010.

[56] H. Paulheim and F. Probst. Application Integration on the User Interface Level: an Ontology-Based Approach. *Data & Knowledge Engineering Journal*, 69(11):1103–1116, 2010.

[57] H. Paulheim and F. Probst. Ontology-Enhanced User Interfaces: A Survey. *International Journal on Semantic Web and Information Systems*, 6(2):36–59, 2010.

[58] A. Polleres and H. Chen, editors. *Proceedings of the ISWC 2010 Posters & Demonstrations Track: Collected Abstracts*, volume 658 of *CEUR-WS*, 2010.

[59] D. Quan, D. Huynh, and D. Karger. Haystack: A Platform for Authoring End User Semantic Web Applications. In D. Fensel, K. P. Sycara, and J. Mylopoulos, editors, *The SemanticWeb - ISWC 2003*, number 2870 in LNCS, pages 738–753. Springer, 2003.

[60] A. Rauschmayer. Semantic-Web-Backed GUI Applications. In *Proceedings of the ISWC 2005 Workshop on End User Semantic Web Interaction*, 2005.

[61] L. Rutledge, m.c. schraefel, A. Bernstein, and D. Degler, editors. *Procedings of the The 3rd International Semantic Web User Interaction Workshop (SWUI06) Workshop*, 2006.

[62] SAP AG. SAP - Components & Tools of SAP Netweaver: SAP Netweaver Portal. `http://www.sap.com/platform/netweaver/components/portal/index.epx`, 2010. Accessed July 20th, 2010.

[63] L. Sauermann, A. Bernardi, and A. Dengel. Overview and Outlook on the Semantic Desktop. In Decker et al. [22].

[64] N. Schmitt, M. Niepert, and H. Stuckenschmidt. BRAMBLE: A Web-based Framework for Interactive RDF-Graph Visualisation. In Polleres and Chen [58].

[65] G. Tummarello, R. Cyganiak, M. Catasta, S. Danielczyk, R. Delbru, and S. Decker. Sig.ma: Live views on the Web of Data. *Web Semantics: Science, Services and Agents on the World Wide Web*, 8(4):355 – 364, 2010. Semantic Web Challenge 2009; User Interaction in Semantic Web research.

[66] W3C. RDF Primer. `http://www.w3.org/TR/rdf-primer/`, February 2004. Accessed September 24, 2010.

[67] W3C. RDF Validation Service. `http://www.w3.org/RDF/Validator/`, 2007. Accessed September 27th, 2010.

[68] C. Wege. Portal Server Technology. *IEEE Internet Computing*, 6(3):73–77, 2002.

[69] W. Weiss, M. Hausenblas, and G. Sprung. Visual Exploration, Query, and Debugging of RDF Graphs. In *Semantic Web User Interaction at CHI 2008*, 2008.

[70] Yahoo! Inc. Pipes: Rewire the web. `http://pipes.yahoo.com/pipes/`, 2010. Accessed July 19th, 2010.

[71] J. Yu, B. Benatallah, F. Casati, and F. Daniel. Understanding Mashup Development. *IEEE Internet Computing*, 12(5):44–52, Sept.-Oct. 2008.