Integration of User Interface Components based on Ontologies and Rules

Heiko Paulheim

SAP Research Center Darmstadt, Germany,
`heiko.paulheim@sap.com`

Abstract. During the past decades, software engineering has changed due to the growing number of available components and the pressure to produce software systems of larger complexity in ever shorter periods of time. The task of integrating components has thus become equally important as developing those components.
Unfortunately, components typically do not come with fitting screws and bolts that make such an integration an easy task. Especially when it comes to the integration of user interface (UI) components, current approaches still lack mechanisms for seamlessly integrating components and allowing cross-component interactions, most notably when heterogeneous components, developed with different technologies and programming languages, are involved. UI integration typically involves the acquisition of knowledge about the applications' internal functionality as well as many hacks and workarounds, which in the end lead to code-tangling and a monolithic architecture that is hard to maintain. This paper discusses an approach employing formal ontologies and rules for overcoming these problems and simplifying UI integration.

1 Introduction

Software applications are typically organized in three layers: the data source layer, the business logic layer, and the user interface layer. Thus, there are three strategies for performing integration of those applications: integration on the database layer, on the business logic layer, and on the user interface layer [1]. The latter has two specific advantages:

1. Existing user interface components can be reused. As developing a user interface consumes about 50% of the overall development time [2], this leads to drastically reduced development efforts.
2. Users are confronted with UI components they already know, instead of having to learn how to operate a user interface which as has been developed from scratch.

Ontologies have been widely used for integration on the data source as well as on the business logic layer [3, 4], solving major issues in system interoperability. However, despite the need of formal models for UI integration has been identified [1], and various approaches to integrate ontologies in UI development exist [5], ontologies have not been employed for UI integration, as depicted in Fig. 1. This paper discusses an ontology-based approach to UI integration.

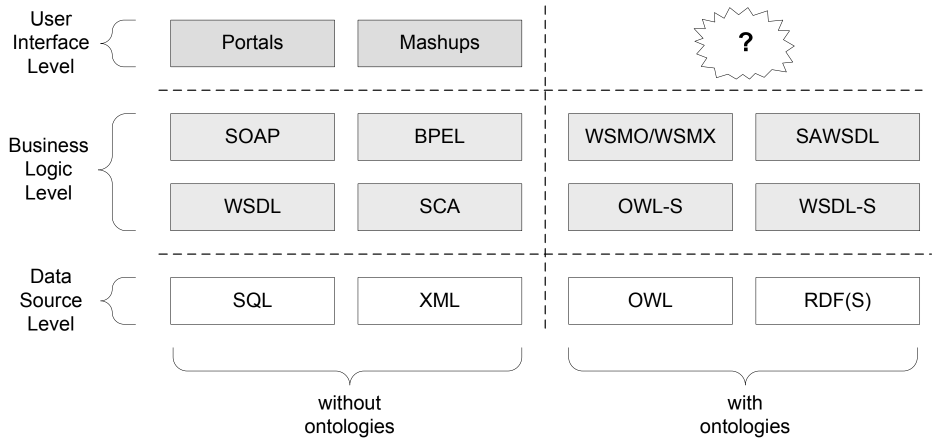

Fig. 1. Examples for integration on the different layers, both with and without ontologies, adapted from [6]. There are currently no approaches employing ontologies on the user interface level.

2 Current Gaps in UI Integration Approaches

Various approaches to UI integration exist, such as portal and mashup tools [7, 8], as well as a couple of research prototypes. Although some of those solutions are quite mature in some fields, e.g., they provide sophisticated tool support and layouting capabilities, most of them share some serious gaps:

– There is no common model of UI components and events. Event exchange is most often based on naming conventions, requiring knowledge about the components' interna.
– Although data can be exchanged in form of XML or JSON, there is no explicit support for converting data between different data models.
– Most approaches are restricted to a certain set of UI technologies, general approaches for implementing seamless integration (e.g., including drag and drop) of heterogeneous UI components are still missing.

The approach discussed in this paper uses ontologies and rules to overcome these gaps. Ontologies can provide a common ground for defining events instead of relying on naming conditions, and they can be used to annotate data objects for facilitating conversion between different data formats. Furthermore, they may provide an abstraction of UI components, which allows for neglecting implementation details of different technological platforms.

3 Approach and Prototype

To show how ontologies can be used in UI integration, we have developed an approach and a Java-based prototype [9]. We use ontologies for formally describing

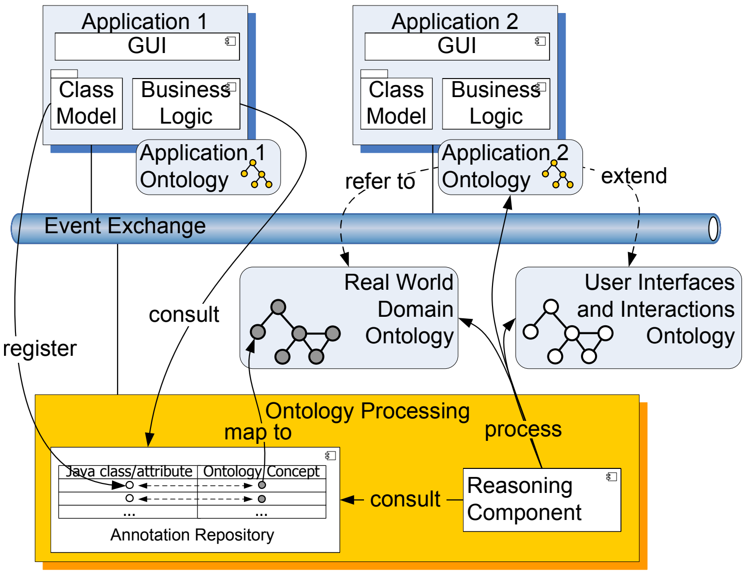

Fig. 2. Prototype architecture.

the UI components to integrate, as well as to annotate events and data objects exchanged. Rules can be defined to coordinate cross-component behavior.

Both annotations of events and objects and integration rules, which are processed by a central reasoner, introduce indirections between the integrated components. Thus, direct dependencies between components can be avoided, which reduces code-tangling and increases the maintainability of the integrated system.

The approach relies on a complete and concise ontology of the user interfaces and interactions domain, which is used in conjunction with a real world domain characterizing the objects processed by the applications, such as customers and bank accounts. Both are used to define application ontologies for the integrated applications. Fig. 2 shows the architecture of the prototype.

4 Preliminary Results

A proof-of-concept implementation of the *SoKNOS* emergency management system [10] has been based on the framework prototype to show the principal applicability of the approach.

By integrating components developed with both Java and Flex, the ability to cross technological borders could be shown. Since the components only communicate based on the annotations of events and data objects, the underlying implementations can be abstracted from. Even seamless integration such as dragging and dropping objects from Flex to Java and vice versa is possible [11].

When dealing with user interfaces, reactivity is a crucial requirement. In a set of systematic experiments, we could show that, by choosing a suitable architecture, the reaction time can be held within appropriate borders, even for a large set of integrated applications [12].

5 Conclusion

Many current UI integration approaches suffer from some significant drawbacks, which hinder the development of maintainable integrated UIs. In this paper, we have shown an approach that uses ontologies and integration rules to remedy those drawbacks.

References

1. Daniel, F., Yu, J., Benatallah, B., Casati, F., Matera, M., Saint-Paul, R.: Understanding UI Integration: A Survey of Problems, Technologies, and Opportunities. IEEE Internet Computing **11**(3) (2007) 59–66
2. Myers, B.A., Rosson, M.B.: Survey on user interface programming. In: CHI '92: Proceedings of the SIGCHI conference on Human factors in computing systems, New York, NY, USA, ACM (1992) 195–202
3. Sahoo, S.S., Halb, W., Hellmann, S., Idehen, K., Jr, T.T., Auer, S., Sequeda, J., Ezzat, A.: A Survey of Current Approaches for Mapping of Relational Databases to RDF. `http://www.w3.org/2005/Incubator/rdb2rdf/RDB2RDF_SurveyReport.pdf` (2009) Accessed July 16th, 2010.
4. de Bruijn, J., Kerrigan, M., Zaremba, M., Fensel, D.: Semantic Web Services. In Staab, S., Studer, R., eds.: Handbook on Ontologies. International Handbooks on Information Systems. 2nd edition edn. Springer (2009) 617–636
5. Paulheim, H., Probst, F.: Ontology-Enhanced User Interfaces: A Survey. International Journal on Semantic Web and Information Systems **6**(2) (2010) 36–59
6. Benatallah, B., Nezhad, H.R.M.: Service Oriented Architecture: Overview and Directions. In Brger, E., Cisternino, A., eds.: Advances in Software Engineering. Volume 5316 of LNCS., Springer (2007) 116–130
7. Wege, C.: Portal Server Technology. IEEE Internet Computing **6**(3) (2002) 73–77
8. Hoyer, V., Fischer, M.: Market Overview of Enterprise Mashup Tools. In: ICSOC '08: Proceedings of the 6th International Conference on Service-Oriented Computing, Berlin, Heidelberg, Springer-Verlag (2008) 708–721
9. Paulheim, H., Probst, F.: Application Integration on the User Interface Level: an Ontology-Based Approach. Data & Knowledge Engineering Journal, Special Issue on Contribution of Ontologies in Designing Advanced Information Systems (2010) to appear.
10. Paulheim, H., Dweling, S., Tso-Sutter, K., Probst, F., Ziegert, T.: Improving Usability of Integrated Emergency Response Systems: The SoKNOS Approach. In: Proceedings "39. Jahrestagung der Gesellschaft fr Informatik e.V. (GI) - Informatik 2009". Volume 154 of LNI. (2009) 1435–1449
11. Paulheim, H., Erdogan, A.: Seamless Integration of Heterogeneous UI Components. In: Proceedings of the 2nd ACM SIGCHI Symposium on Engineering Interactive Computing Systems (EICS 2010), ACM (2010) 303–308
12. Paulheim, H.: Efficient Semantic Event Processing: Lessons Learned in User Interface Integration. In Aroyo, L., Antoniou, G., Hyvönen, E., ten Teije, A., Stuckenschmidt, H., Cabral, L., Tudorache, T., eds.: The Semantic Web: Research and Applications (ESWC 2010), Part II. Volume 6089 of LNCS., Springer (2010) 60–74