

Semantic Desktop Systems for Context Awareness – Requirements and Architectural Implications

Simone Braun, Andreas Schmidt, Christina Hentschel

FZI Research Center for Information Technologies
Information Process Engineering
Haid-und-Neu-Straße 10-14, 76131 Karlsruhe, Germany
{Simone.Braun|Andreas.Schmidt|Christina.Hentschel}@fzi.de

Abstract. Semantic Desktop systems appear to be a promising infrastructure for context-aware applications that acquire and make use of user context information in order to provide more efficient interaction between the user and the system. In a systematic analysis, we have analyzed existing Semantic Desktop systems based on a collection of requirements and deduced from it architectural implications for future Semantic Desktop development.

1 Introduction

Semantic Desktop systems have emerged as a promising approach to enhance desktop work environments; especially of knowledge workers. This development is deeply rooted in the conviction that the current environments for computer-based work are far from adequate in their support for complex, interrelated, and quickly changing tasks. Another major development are context-aware systems, originating from the field of ubiquitous computing, which are now moving towards desktop applications; e.g., in the area of supporting workplace learning ([1], [2]). These systems attempt to capture aspects of the current situation of the user (including the personal state of the user and the environment) in order to improve the efficiency of interaction with the system. This could include improving precision of search results, proactive recommendations, mediation of communication, among others.

Semantic Desktop systems already capture a relevant portion of the user's context by interfacing with the conventional applications (office applications, personal information managers, file systems, messenger applications etc.). Instead of duplicating this, it would be natural to leverage on work in the Semantic Desktop field in order to simplify the development of context-aware applications. In this paper, we want to present requirements for context-awareness on top of Semantic Desktop systems (section 2), a systematic analysis of current Semantic Desktop systems with respect to those requirements (section 3), and directions in terms of architecture for future Semantic Desktop developments (section 5).

2 Requirements Specification

In a first step, we characterize what context is and what it should encompass as features. Then we elaborate the requirements of a semantic desktop infrastructure providing context information.

2.1 Characterizing Context

Before we can establish requirements for context-awareness on top of Semantic Desktop systems, we need to characterize what “context” is (cf. [3], [4]). We based our characterization on work on general user modeling [5], personal knowledge management (e.g., [6]), and our prior work on context-steered workplace learning (cf. [7], [1]), on which we want to revert as main guiding scenario.

The main goal of context-steered workplace learning is to integrate and support learning processes within work processes. Instead of having employees actively searching for appropriate learning material, the learning support system observes what the employee is currently doing. Based on these observations and background knowledge, e.g. on competence requirements on the employee’s task, business process etc., the system deduces the possible knowledge gap and recommends appropriate learning resources. Furthermore, the system can also suggest other persons, e.g. co-workers, who are experts in a certain area, or who recently experienced a similar situation and whom the employee can approach.

In this scenario, context-awareness has many different aspects; whereas the *What* (e.g. user competences vs. task requirements, trust vs. expertise), the *When* (e.g. at task beginning, not before a meeting, availability of contact person), and the *How* (e.g. required attention, notification preferences) of delivering learning resources/contact persons are in focus. Fig. 1 shows the resulting context features, which characterize the user context. These context features can be divided into seven categories: personal, social, temporal, environmental, technical, organizational and operational features:

- **Personal features** are characteristics of the user herself including her competences, preferences and habits like daily routines. Thus, questions, for instance, on a user’s knowledge about a specific topic and potential knowledge gaps can be answered.
- **Social features** give information about social contacts of the user. That means the quality of social relationships towards other people. For instance, it is less awkward to ask a confided colleague for help than a rival. This quality of social relationships is described by roles, trust and intensity.
- **Temporal features** are date and time; e.g. in order to determine when to deliver learning resources.
- **Environmental features** describe the user’s location.
- **Technical features** subsume technical characteristics of the user’s computer. These are available applications, e.g. email or calendar, and available resources, e.g. file system, and (organizational) data repositories etc. They help in determining, for instance, if the user is currently absent, thus cannot act as contact person, or to specify learning relevant documents.

Competences Preferences Habits	Role Trust Intensity	Date Time	Location
Personal	Social	Temporal	Environmental
User Context Features			
Technical	Organizational	Operational	
Available Applications Available Resources	Organizational Unit Job Position Current Project Current Business Process Current Task	Open Applications Active Application Open Resources User Actions Attention Time Capacity	

Fig. 1. User context features

- **Organizational features** give information about the user’s placement within the organization. That includes the organizational unit, e.g. division or group of a company, the job position, the current project the user is working in, or the current business process and task the user is in. This indicates, for instance, the required knowledge.
- **Operational features** describe detailed user activities. These are open applications and the currently active application and open resources meaning concrete documents or application data like an email. Based on these three features but more detailed are the user actions like creating, reading, writing, modifying or closing a document. Another operational feature is the user’s attention she pays to a resource or application and more specific to which part of. That allows to specify how long a document is open and which part the user take a look at. The last feature is the user’s time capacity; for instance, how much time does the user have until the next meeting. Depending on that, the user eventually needs more condensed information.

2.2 Requirements on the Semantic Desktop Infrastructure

Using Semantic Desktops as infrastructure and user context provider for context-aware applications implicates specific requirements, Semantic Desktops need to fulfill. We identified eight requirements presented in detail in the following:

1. **Provision of subsets of the current context.** For enabling applications, which offer, for instance, learning objects in real-time, we need access to the user’s current context information like her current task or time capacity. At the same time, such

an application does not need all context information; e.g. about social relationships. Filtering the wanted information would be cost- and time-consuming. Thus, Semantic Desktops should allow the provision of subsets of the current context so that only wanted information can be requested.

2. **Well-defined API.** In order to make use of the context information, it is necessary to have a well-defined API that allows the access for external applications. Otherwise, the performance and functionality of context-aware applications cannot be guaranteed in case the Semantic Desktop is modified, e.g. extended by additional resources or applications.
3. **Asynchronous notification of context changes.** The context of an user is never static but changes continuously. For enabling proactive behavior of context-aware applications, e.g. the recommendation of learning objects without the need for explicit user request, the Semantic Desktop has to notify (interested) context-aware applications of context changes.
4. **Provision of historical context information.** Context-aware applications do not only need information about the user's current context but also about its history. Historical context information, e.g. user actions, allow for recognizing patterns in user behavior or inferring new information, e.g. daily routines, which can be used for the prediction of future user actions. Consequently, Semantic Desktops need to store context information for later use and must not overwrite it with the current information.
5. **Consideration of uncertainty.** Semantic Desktops collect context information typically automatically and via indirect methods; e.g. by observing user behavior and inferring facts from these observations. However, the results are not always certain. They vary in their probability, precision and consistency. For instance, no calendar entries do not imply a person's availability. There can be spontaneous meetings. Or in turn, a declined but not signed out calendar entry does not imply a person's absence. Thus, the information is uncertain and that has to be assumed in general. Semantic Desktops have to consider this uncertainty, e.g. by specifying a degree of uncertainty.
6. **Access to the context information of other users.** Another requirement is the access to context information of other users. If we consider real-time learning situations at the workplace, a context-aware application could not only recommend documents for support but also other persons whom to ask for help. Therefore, information about their context, e.g. competences or availability, is necessary.
7. **Consideration of dynamics.** Semantic Desktops do not only need to consider context changes for notifying context-aware applications but also its dynamics. That means how it changes. Collected context information is not valid indefinitely. If the system gets to know about the current task of the user, this information will only be valid for a limited amount of time. Furthermore different features of the context change at different pace; e.g., competences or job position change less frequently than location or current task. Therefore it is necessary that Semantic Desktops specify, for instance, a validity period or deadline for the context information.

3 Analysis of Existing Systems

Based on these presented requirements for context-awareness, we examined in our analysis how existing Semantic Desktop systems are suited for infrastructure and user context provider for context-aware applications. With reference to these requirements, we regarded three main aspects addressing the content of context information, interfaces for accessing the information and the system architecture. We analysed five systems¹: Haystack by the MIT [8,9,10,11], Gnowsis by the DFKI [12,13,14,15], D-BIN by SE-MEDIA [16,17,18], Iris by SRI [19,20] and Chandler by the OSA foundation [21]. Their publications and documentations served as a basis for our study. Due to time constraints, we could not look at the source code in detail.

Features	Haystack	Gnowsis	DBin	IRIS
<i>Personal</i>				
Competences	1	1,*	1	1
Preferences	5,1	1,*	1,*	1,*
Habits	1	1	1	1
<i>Social</i>				
Role	1	1,*	1	1
Trust	1	1	1	1
Intensity	1	1	1	1
<i>Temporal</i>				
Date	5	5	5	5
Time	5	5	5	5
<i>Environmental</i>				
Location	1,*	3-5	1	1,*
<i>Technical</i>				
Available Applications	5	5	5	5
Available Resources	5	5	5	5
<i>Organizational</i>				
Organizational Unit	1	3-5	1	1
Job Position	1	3-5	1	1
Current Project	1	5	1	*
Current Business Process	1	3	1	1
Current Task	2	4	1	1
<i>Operational</i>				
Open Applications	-	3-5	-	-
Active Application	5	3-5	-	5
Open Resources	5	3-5	1	5
User Actions	5	5	1	5
Attention	1,*	3-5	1	1,*
Time Capacity	1,*	1,*	1	1,*

Fig. 2. Evaluation of the first requirement concerning the context features in detail (1:undefined, 2:planned, 3:conceptual, 4:prototypical, 5:realized, *:applicable)

¹ see <http://www.semanticdesktop.org/> for an overview

For the evaluation of the requirements, we differentiate between the following ranges:

1: undefined. No information available or plans abandoned

2: planed. With the aim of implementing the requirement

3: conceptual. Approach/model already developed

4: prototypical. Prototypical implementation available

5: realized. Implementation completed

***: applicable.** No information given, but implementation easily possible

Fig. 2 and Fig. 3 show the detailed evaluation results. If no distinct classification is possible, a span of ranges will be given, respectively a list in case of partial provision/implementation.

Due to its early development stage, its only limited documentation, and because comparability with the other systems would have not been given, we disregarded from evaluating the Chandler system. Nevertheless, its conceptual approach looks promising and should be examined at a later stage.

Requirements	Haystack	Gnowsis	DBin	IRIS
Well-defined API	5	5	5	5
Provision of subsets of the current context	1	5	1	1
Asynchronous notification of context changes	5	5,*	1	5
Access to the context information of other users	1	1	5	2
Provision of historical context information	5,*	3-5,*	*	*
Consideration of uncertainty	1	5	1	1
Consideration of dynamics	*	1	1	1

Fig. 3. Evaluation of 2.-8. requirement (1:undefined, 2:planed, 3:conceptual, 4:prototypical, 5:realized, *:applicable)

3.1 Summing-up the Results

Haystack stands out because of its possibilities for personalization as well as for supporting the dynamics of context information. The user can extend the data model by defining own meta data or she can define own views and operations. These personalization possibilities can be used as indicators for an user's preferences. Dynamics could be supported because of so-called statements, which are further annotated with meta data like the author. If there is inconsistent information, the user can decide, which to use in the future and which to discard. It was planned but then abandoned that services delete

old statements and replace them with the current one automatically. Relating to the provision of historical context information, thus Haystack displays data, the user accessed earlier. That means, information about open data resources is available. For providing this functionality, the history of user operation (actions) is used. As a consequence, these are available, too, as well as a browsing history. Further historical information is not explicitly mentioned but can easily be implemented. However, the system lacks of integrating cross-organizational data repositories so that background knowledge is not available, which helps determining organizational features, competences or social features.

By contrast, Gnowsis can fall back on such knowledge. That is the reason for its good results concerning the context features. Overall, Gnowsis fulfills most of the requirements. It is furthermore the only system that considers uncertainty by providing an uncertainty value. That means, directly observed context information get a value of 1.0, whereas deduced information get a lower value. Furthermore, information about creation of context information and their inference are available. However, in some cases it is ambiguous which approaches are only conceptual or already implemented, e.g. concerning historical context information.

DBin got the worst results. As the Haystack system, its approach is to completely replace the usual desktop with all its applications and data sources and to provide one holistic system. However, the DBin system only integrates the file system at the moment. This, together with often only sparsely available information, causes DBin's poorer results. Anyhow, DBin stands out positively because of the access to context information of other users. Users can make their own context information public. If another user is interested in, the data are automatically exchanged.

IRIS got similar results as the Haystack system. An outstanding feature of IRIS is its framework for machine learning that enables the inference of new context information. But compared with Haystack, we could not find any information about the provision of historical context information or the consideration of dynamics for the IRIS system.

Altogether, none of the Semantic Desktop systems really considers the dynamics of context information. With the exception of the Haystack system where possibilities for supporting dynamics are given. Similarly, the systems do not take into account uncertainty aspects of context information except for the Gnowsis system. The provision of historical context information, in particular of user actions, exists partially or is at least applicable and easy to implement. The access of context information of other users once planned and once implemented, but else undefined. Every system provides a well-defined API. However, only Gnowsis depicts the query language.

With regard to the context features, an evaluation for open applications for Haystack and IRIS is not meaningful because they integrate all applications in their user interface so that these are all always open. Thus, a differentiation between open or closed applications is not possible. Concerning DBin, it does not integrate any application than the file system. Thus a rating is not possible, too. Overall, it stands out that the personal and social context features are hardly supported (cf. red marked in Fig. 4). But there are good potentials where the Semantic Desktop systems integrate cross-organizational data resources and applications or where they connect the individual desktops. When cross-organizational data are well maintained, it is not far too hard to determine con-

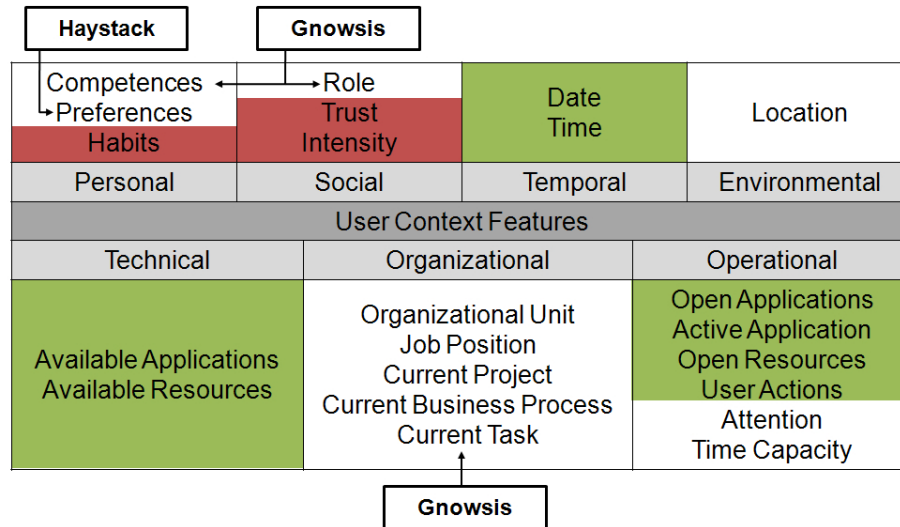


Fig. 4. Overview of the fulfillment of the context features

text features such as competences, preferences or role. Similarly, background knowledge about the organization facilitates the determination of the organizational context features and that the Gnowsis system is availing. Furthermore, all systems provide information for the temporal and technical context features (cf. green marked in Fig. 4). The first four operational features are consistently well supported and there are also possibilities for the attention and time capacity features, except for the DBin system.

4 Implications for Future Semantic Desktop Architectures

As the analysis has shown, there *are* potentials for using Semantic Desktop systems as context sources for context-aware systems. But as they were designed with a different purpose in mind, they are not geared towards dealing with the peculiarities of user context information [22]. In this section, we want to briefly sketch how the consideration of user context affects the architectural development of future Semantic Desktop systems:

- **Layered Architecture.** As presented in section 3, current systems can mainly provide context information related to personal information management tasks. However, context as presented needs a much wider scope, e.g., by incorporating the organizational context, which is usually stored in organizational systems. This goes well beyond the focus of Semantic Desktop systems, so it is not necessarily advisable to make them a container for everything. But still, the information extraction components that lift traditional application data and events to the semantic level can be quite useful (e.g., for using the file system or Outlook). On the user side, the user context management component does not need to build on the user interface of the Semantic Desktop so that a three layer architecture is recommended:

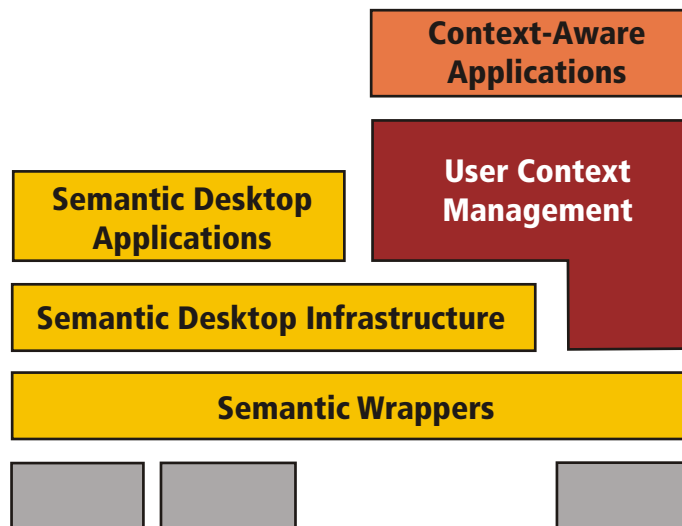


Fig. 5. Architecture

semantic wrappers at the lowermost level, semantic desktop infrastructure as the core, non-UI services, and semantic desktop application at the top, interacting directly with the user (see Fig. 5). That way, we can reuse functionality as much as possible without losing the focus of development. This is, by the way, in line with current developments in the NEPOMUK EU project² where a similar three-layer architecture is proposed.

- **Mixture of Virtual and Materialized Integration.** Currently, Semantic Desktop systems like Gnowsis mainly are built according to the paradigm of materialized integration ([23]), i.e., existing sources are crawled and the result is stored in a central RDF storage. This becomes a limiting factor as soon as we need to consider large amounts of data from which we only need to consider a small portion or if we have to consider the dynamics of context information. This especially affects the *Semantic Wrapper* layer which needs to add query capabilities.
- **Imperfection perspective.** Lifting conventional applications to the semantic level often involves heuristics (e.g., for solving object identity issues, for classifying documents etc.). This information is used within the user context manager for complex augmentation procedures and reasoning, which in itself is only heuristic in many aspects. If we do not consider the reliability of this process, it can lead to unusable results. The integration process of the Semantic Wrappers and the Semantic Desktop Infrastructure needs to explicitly represent confidence of delivered information so that the user context management service can take this information into account (as show in [22]).

² <http://nepomuk.semanticdesktop.org/>

- **Temporal perspective and consideration of dynamics.** For deriving user context features from lower level events, it is crucial that we do not only have to rely on the *current* context. For augmenting and aggregating user context information, we need historical data about a user. This has a huge impact on the amount of context data we are dealing with. This in turn means that the Semantic Desktop infrastructures stores its information with timestamp information in order to exploit for example linkage information. Furthermore, if we consider more dynamic context information like the current task which can probably not be provided as a continuous feed, we also need to signal for how long this information is valid (by specifying a validity interval).
- **Access to the context information of others.** A critical area is the access to the context information of others. On the one hand, this is very important if we want to stimulate exchange between different users (e.g., for recommending informal learning opportunities with others who did similar things recently). On the other hand, context information is very sensitive and we need to deal carefully with privacy issues. While the first generation of Semantic Desktop systems was not able to communicate with others, the Social Semantic Desktop idea gains ground and enables a peer-to-peer interchange of resources. However, this does currently not include much information about the user herself so that also the sharing mechanisms are not sophisticated in terms of access rights.

5 Conclusions

We have presented a systematic analysis of requirements and current suitability of Semantic Desktop systems as a basis for context-aware applications. The consideration of context-awareness aspects within Semantic Desktop systems would broaden the scope of Semantic Desktop systems towards a universal desktop infrastructure for advanced applications, as has been shown with an early extension of Gnowsis [6]. However, a more large-scale approach requires also the consideration of peculiarities of user context information within the architecture of the semantic desktop infrastructure. Current developments in the NEPOMUK project towards a layered architecture of Semantic Desktop systems fit quite well into this direction, but the interfaces of the different components need to take into account temporality, imperfection and integration paradigm issues. Here, Semantic Desktop development should investigate into the body of research around scalable and flexible information integration (e.g., [24]).

Acknowledgements. This work was supported by the German Federal Ministry for Education and Research within the project *Im Wissensnetz*³.

References

1. Schmidt, A., Braun, S.: Context-aware workplace learning support: Concept, experiences, and remaining challenges. In Nejdil, W., Tochtermann, K., eds.: First European Conference

³ <http://www.im-wissensnetz.de>

- on Technology Enhanced Learning - EC-TEL 2006, Berlin Heidelberg, Germany, Springer-Verlag (2006) 518–524
2. Lindstaedt, S.N., Mayer, H.: A Storyboard of the APOSDLE Vision. In Nejd, W., Tochtermann, K., eds.: *Innovative Approaches for Learning and Knowledge Sharing. Proceedings of the First European Conference on Technology-Enhanced Learning (ECTEL 06)*. Volume 4227 of LNCS., Berlin Heidelberg, Germany, Springer-Verlag (2006) 628–633
 3. Dey, A.K.: Understanding and using context. *Personal and Ubiquitous Computing Journal* **1** (2001) 4–7
 4. Dourish, P.: What we talk about when we talk about context. *Personal Ubiquitous Computing* **8** (2004) 19–30
 5. Heckmann, D., Schwartz, T., Brandherm, B., Schmitz, M., von Wilamowitz-Moellendorff, M.: Gumo – the general user model ontology. In: *10th International Conference on User Modeling (UM 05)*, Edinburgh. (2005)
 6. Schwarz, S.: A context model for personal knowledge management. In: *Proceedings of the IJCAI-05 Workshop on Modeling and Retrieval of Context Edinburgh, July 31 - August 1, 2005. CEUR Workshop Proceedings* (2005)
 7. Schmidt, A.: Potentials and challenges of context awareness for learning solutions. In: *LWA 2005: Lernen–Wissensentdeckung–Adaptivität, 13th Annual Workshop of the SIG Adaptivity and User Modeling in Interactive Systems (ABIS 2005)*, Saarbrücken. (2005)
 8. Haystack Project. <http://haystack.lcs.mit.edu/> (2007) (accessed 2007-04-04).
 9. Adar, E., Kargar, D., Stein, L.A.: Haystack: Per-user Information Environments. In: *CIKM '99: Proceedings of the eighth international conference on Information and knowledge management*, New York, NY, USA, ACM Press (1999) 413–422
 10. Karger, D.R., Bakshi, K., Huynh, D., Quan, D., Sinha, V.: Haystack: A Customizable General-Purpose Information Management Tool for End Users of Semistructured Data. In: *CIDR*. (2005)
 11. Quan, D., Huynh, D., Karger, D.R.: Haystack: A Platform for Authoring End User Semantic Web Applications. In: *Second International Semantic Web Conference (ISWC 2003)*. (2003) 738–753
 12. Gnowsis. <http://www.gnowsis.org/> (2007) (accessed 2007-04-04).
 13. Sauer mann, L.: The gnowsis semantic desktop for information integration. In: *Proceedings of the WM 2005*, Springer (2005)
 14. Sauer mann, L.: The semantic desktop - a basis for personal knowledge management. In Maurer, H., Calude, C., Salomaa, A., Tochtermann, K., eds.: *Proceedings of the I-KNOW 2005. 5th International Conference on Knowledge Management*. (2005)
 15. Sauer mann, L., Grimnes, G.A., Kiesel, M., Fluit, C., Maus, H., Heim, D., Nadeem, D., Horak, B., Dengel, A.: Semantic desktop 2.0: The gnowsis experience. In Cruz, I.F., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L., eds.: *International Semantic Web Conference. Volume 4273 of Lecture Notes in Computer Science.*, Springer (2006) 887–900
 16. DBin. <http://dbin.org/> (2007) (accessed 2007-04-04).
 17. Tummarello, G., Morbidoni, C., Puliti, P., Piazza, F.: The dbin semantic web platform: an overview. In: *Proceedings of WWW2005*. (2005)
 18. Tummarello, G., Morbidoni, C., Nucci, M.: Enabling semantic web communities with dbin: An overview. In Cruz, I.F., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L., eds.: *International Semantic Web Conference. Volume 4273 of Lecture Notes in Computer Science.*, Springer (2006) 943–950
 19. IRIS. <http://www.openiris.org/> (2007) (accessed 2007-04-04).
 20. Cheyer, A., Park, J., Giuli, R.: Iris: Integrate. relate. infer. share. 1st Workshop on The Semantic Desktop. 4th International Semantic Web Conference (2005)

21. Chandler. <http://chandler.osafoundation.org/> (2007) (accessed 2007-04-04).
22. Schmidt, A.: A layered model for user context management with controlled aging and imperfection handling. In Roth-Berghofer, T.R., Schulz, S., Leake, D.B., eds.: Modeling and Retrieval of Context. Proceedings of the 2nd International Workshop on Modeling and Retrieval of Context MRC 2005, Edinburgh, Scotland, July 31 - August 1, 2005. Volume 3946 of Lecture Notes in Artificial Intelligence., Springer (2006) 86–100
23. Widom, J.: Integrating heterogeneous databases: Lazy or eager? *ACM Computing Surveys* **28** (1996)
24. Abiteboul, S., Buneman, P., Suciu, D.: *Data on the Web: from relations to semistructured data and XML*. Morgan Kaufmann (1999)