A Constraint-Based Graph Visualisation Architecture for Mobile Semantic Web Interfaces

Daniel Sonntag and Philipp Heim

German Research Center for Artificial Intelligence
66123 Saarbrücken, Germany
sonntag@dfki.de, heim@interactivesystems.info

Abstract. Multimodal and dialogue-based mobile interfaces to the Semantic Web offer access to complex knowledge and information structures. We explore more fine-grained co-ordination of multimodal presentations in mobile environments by graph visualisations and navigation in ontological RDF result structures and multimedia archives. Semantic Navigation employs integrated ontology structures and leverages graphical user interface activity for dialogical interaction on mobile devices. Hence information visualisation benefits from the Semantic Web. Constraint-based programming helps to find optimised multimedia graph visualisations. We report on the constraint-formulisation process to optimise the visualisation of semantic-based information on small devices and its integration in a distributed dialogue system.

1 Introduction

For every specific type of information there are certain categories of visual representations that are more suitable than others. The use of a graph for the visualisation of information has the advantage that it can capture a detailed knowledge structure. Therefore graphs are suitable for conveying semantic relations between individual information items and for providing an understanding of the overall information structure. We aim to display information that stem from semantic RDF[1] structures and explore if the implicit graph structure in the RDF data can be used for the knowledge visualisation process, especially on mobile devices. By additional graph presentations of answers in a linguistic question answering scenario, the user would become more engaged in the dialogue, navigate through the incrementally presented result space, and would be encouraged to pose follow-up questions in natural language. The challenge we address is the intuitive navigation in a semantically organised information space on small interaction devices. Using RDF structures for graph representations can improve the users' understanding of certain information pieces and the relations between these pieces. The second aim is to produce evidence for this hypothesis by implementing and evaluating mobile Semantic Web interfaces and applying direct structure mapping from RDF graphs toward their multimedia visualisations.

[1] Resource Description Framework, *http://www.w3.org/RDF/*

Fig. 1. SmartWeb's main graphical user interface (left) and semantic navigation interactions (centre and right). By clicking on a certain vertex of the graph, the user can change the *focus point* for the fisheye distortion. With a click on the red arrow, the user can change an *active instance*. Clicking again on an active instance is an interaction form to ask for additional detailed multimodal information. When the structure of the dynamic graph changes, a new optimal layout is computed server-side. A further click on the *Ergebnis* (result) node results in displaying the information: **5:3** n. E., 1:1 n. V. (**1:1**, **0:0**), *Ereignis* (incidence) reveals red card for player *Cufre*, for example.

In our most recent dialogue system project SmartWeb [1], we try to provide intuitive multimodal access to a rich selection of Web-based information services; especially the handheld scenario is tailored toward multimodal interaction with ontological knowledge bases and Semantic Web services [2]. Since the application domain we have in mind is football, the knowledge base covers facts about football events, players, matches, etc., the user can ask for. The main scenario[2] we modelled is that a football fan is in Berlin to visit the 2006 FIFA World Cup. Holding the personal digital assistant (PDA) in one hand, she could, for example, ask questions like *How many goals has Michael Ballack scored this year?* or *How did Germany play against Argentina in the FIFA worldcup?* The summarised answer to the last question, SmartWeb provides and synthesises, is *5 Spiele* (5 matches). This is presented along with textual material *Argentinien-Deutschland, (1:3) 8.6.1958 Gruppenspiel (group match), (0:0) 16.7.1966 Grup-*

[2] A scenario presentation provided by Deutsche Telekom can be downloaded at *http://smartweb.dfki.de/SmartWeb_FlashDemo_eng_v09.exe*

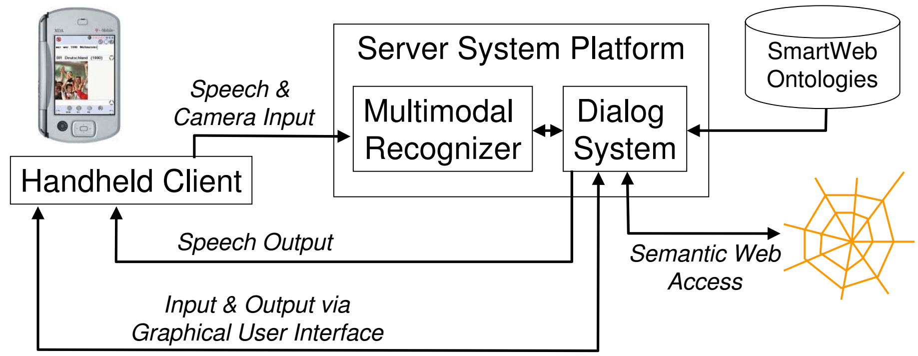


Fig. 2. The distributed SmartWeb dialogue system architecture with handheld client.

penspiel, (3:2) 29.6.1986 Finale (1:0) 8.7.1990 Finale, (5:3) 30.6.2006 Viertelfinale (quarter final), and multimedia material such as images and videos, as can be seen in figure 1. Figure 1 shows the PDA interaction device, the graphical user interface [3], and the semantic navigation possibilities for manipulative interfaces with overview, zoom, filter, and details on demand functionality. To offer an overview and details on demand on small screen size at the same time, we use a fisheye distortion view in combination with automatic graph node placement.

The partners of the SmartWeb project share experience from earlier dialogue system projects [4–7]. We followed guidelines for multimodal interaction, as explained in [8] for example, in the development process of our first demonstrator system [9] which contains the following assets: *multimodality*, more modalities allow for more natural communication, *encapsulation*, we encapsulate the multimodal dialogue interface proper from the application, *standards*, adopting to standards opens the door to scalability, since we can re-use ours as well as other's resources, and *representation*. A shared representation and a common ontological knowledge base eases the data flow among components and avoids costly transformation processes [10], which applies to the visualisation process, too. The general SmartWeb handheld architecture to meet the requirements can be seen in figure 2, the dialogue system as well as the multimodal speech and camera recogniser (for face orientation) are explained in further depth in [2].

In this contribution we report on the use of RDF metadata to arrange information pieces in automatically layout graphs with respect to their semantic relations extracted from RDF results obtained from our knowledge servers. Humans themselves may encode information based upon its meaning [11], at least users feel familiar with this way of information arrangement. The text is structured as follows: chapter 2 and 3 describe the interaction and navigation possibilities, in chapter 4 we report on the integration process, chapter 5 presents related work, and in chapter 6 we conclude by further motivating the use of ontologies and Semantic Web data structures [12] for multimodal interaction design and implementation, and in particular, for visualising graph-like information spaces on mobile PDA devices. Our graph visualisation should provide an answer to the question how conceptual data models facilitate the generation of semantic navigation structures on mobile devices.

2 Interaction Possibilities

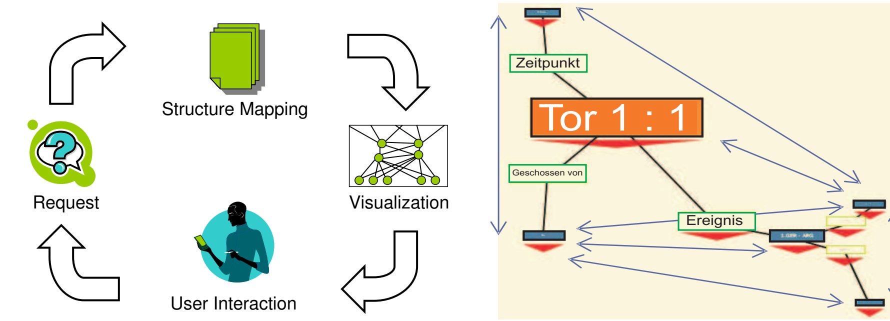
Basically, we want to allow the user to send requests to various information services that are linked by a Semantic Web framework. The user should be able to pose questions in natural language to get multimodal answers to be presented. Subsequently, semantic navigation (section 3) allows for iterative information retrieval by browsing and navigating through self-organising and highly interactive graph structures—as multimodal dialogue system functionality. We group the interaction modalities into three major classes:

- auditory: speech input and output
- graphical: all modalities that serve as input on the screen: touch or stylus input, the keyboard, and for output the graphical display itself
- haptic: device buttons and the cursor joystick

We focus on the touch screen stylus input and the graphical display output. The active graph node, *1.GER-ARG* in figure 1(centre), is called the *focus point*. All direct node interaction possibilities, such as changing an active ontology instance (e.g. a football game instance), can be done on the focus point. The focus point covers multiple similar ontology instances (e.g. of the same type), according to the calculated best mapping from the result structure toward the visual graph structure. Every ontology instance has information slots (figure 1(right)) and relations to other ontology instances which are represented by relation nodes. We calculate the best initial set of active relations. In addition, the user should be able to control which node relations are active and which are not. Starting from this initial setting, the user can change the focus point by clicking other instance or relation nodes.

3 Semantic Navigation

In [13] the use of multiple and distinct ontologies to support modelling, integration, and visualisation of personalised knowledge is discussed. *Semantic Navigation* is thereby defined as a way to build up and navigate views according to the logical organisation given by topic ontologies. This definition accentuates the need for content visualisation and navigation in heterogeneous ontologies, and for authoring or extraction needs. In the context of mobile interfaces and browsing in ontological answer structures, we focus on semantic navigation that helps to (1) access semantic information quickly, (2) allow for intuitive interaction, (3) allow the user to build an own cognitive map due to dynamic exploration of an unknown information space through graph interactions on mobile device displays. To reach this goal the RDF data must somehow be filtered, simplified,



(a) (b)

Fig. 3. (a) Interaction and visualisation loop. (b) Uniform vertices distribution.

and summarised to be appropriate for the display on small devices and navigation. In addition, each graph to be displayed and its layout must be calculated according to the representation and presentation constraints, named *structure mapping* and *visualisation*, respectively. Figure 3(a) shows the interaction and visualisation loop. For example, examining the centre display of figure 1, every exchange of the active instance as user interaction initiates a new interaction and visualisation loop to calculate a new graph and its layout. In our framework we rely on an integrated ontology model where the visual graph can be extracted from result RDF structures in a straightforward way. We use all domain instances according to the ontology assertions, whereby structure mapping reduces the number of nodes needed for representing the RDF data to better fit on the small screen. The instances are grouped according to their incoming and outgoing slots (contextual structure), and every group is represented by one graph node. By sharing the same contextual structure, groups of instances provide the capabilities to be represented by one node in the graph structure. [3]

3.1 Automatic Graph Structures and Graph Layout

Central idea of this approach is to address automated layout by using constraint processing techniques [14] to represent and process causal design principles and perceptual aesthetic criteria about human visual abilities for structuring and organising information. The three central constraints for our automatic graphs are:

1. All vertices must be inside a fixed space on the handheld.
2. Vertices must not overlap.

[3] In the integrated ontology framework this means to specify PDA slot preferences for e.g. match instances: (i) *referee*, (ii) *weather*, (iii) *spectators*, and so on (figure 1(right)). The constraint-based visualisation step optimises the layout in such a way that five ontological soccer match instances are grouped into one graph node (figure 1(middle)).

3. Related vertices must be placed next to each other.

In addition to these constraints for the automatic graph layout, a number of aesthetic criteria should be considered as far as possible [15]. For display on a small mobile device, we take four aesthetic criteria into consideration:

1. Avoid edge crossings.
2. Keep edge lengths uniform.
3. Distribute vertices uniformly.
4. Keep vertices conform with user expectations.[4]

The design of an aesthetically pleasing layout is characterised as a combination of a general search problem in a finite discrete search space and a constraint satisfaction optimisation problem (CSP) [16]. The task is to find positions for all instance vertices satisfying the three constraints, as well as the four aesthetic criteria which should be considered as far as possible. The computation of the solution has to be fast to guarantee real-time interactivity.[5]

A CSP is defined by a set of variables, $X_1, X_2, ..., X_n$, and a set of constraints, $C_1, C_2, ..., C_m$, whereby each variable X_i has a nonempty domain D_i of possible values. Each constraint C_i involves some subset of the variables and specifies the allowable combinations of values for that subset. A CSP state is defined by an assignment of values to some or all of the variables, $\{X_i = v_i, X_j = v_j, ...\}$. In a *complete assignment* every variable is mentioned, and a CSP *solution* is a complete assignment that satisfies all the constraints. Fortunately, the problem to satisfy the constraints mentioned above can be modelled by a simple kind of CSPs; it involves only discrete variables that have finite domains for vertex positions. A position consists of discrete x- and y- coordinate numbers. The domains for these coordinates are restricted by the width and height of the handheld display. If the maximum domain size of any variable in a CSP is d, then the number of possible complete assignments is $O(d^n)$, where n is the number of variables. In the worst case, therefore, we cannot expect to solve finite-domain CSPs in less than exponential time that is, exponential in the number of variables, but fortunately the number of variables (nodes) is rather low in our domain. We chose a *refinement model*[6] over an *perturbation model*[7],

[4] To provide consistent visual encodings and smooth transitions between consecutive displays.

[5] A time limit of three seconds on producing a satisfying layout solution is set. If the constraints are not satisfied within that limit, we assume the constraints to be inconsistent. A way to handle inconsistent constraints is described in section 3.2.

[6] Variables are initially unconstrained; constraints are added as the computation unfolds, progressively refining the permissible values of the variables. Solving CSPs is based on removing inconsistent values from variables' domains until the solution is found (by *forward checking*, to look at each unassigned variable Y that is connected to X by a constraint and deleting from Y 's domain any value that is inconsistent with the value chosen for X. If a partial solution violates any of the constraints, backtracking is performed.

[7] At the beginning of an execution cycle variables have specific values associated with them that satisfy the constraints. The values of one or more variables are perturbed,

because the latter corresponds to a local search method, that can effectively use previous graph CSP solutions, but biases local refinements without explicit statement, and because we are interested in a more principled approach with control over most of the declarative constraints. We use the Choco Constraint Programming System[8] as refinement model API. Choco provides a Java library for CSPs built on an event-based propagation mechanism with backtrackable structures.

Constraints Formulation. First we code the vertex positions into a suitable representation for constraint formulation. Since the Choco system does not support tuple structures we divided a vertex position into two constraint variables. The distance between two vertex positions P_1 and P_2 is formulated as constraints for distances between two x-coordinates (x_1, x_2) and two y-coordinates (y_1, y_2). The domain of possible discrete values for these variables is limited to the fixed space on the handheld reserved for the graph presentation (480x600 pixels). For all pairs of vertices we formulate a distance constraint C_1 to avoid the overlapping of two or more vertices. This minimum distance constraint prevents overlapping by setting a minimum separation distance value between all vertices. As a second distance constraint C_2 in the opposite direction, we formulate a maximal distance constraint to place vertices next to each other for all pairs of related vertices. The distance between two 2D positions is normally computed as Euclidean distance $distance = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$. Because neither power nor radical operators available in the Choco system due to complexity and performance reasons, we used distance approximations by elementary calculation types, like addition and subtraction. We experimented with a simple approximation derived from the Manhattan distance ($L_1 norm$), where the distance is computed as:

$$distance = |x_1 - x_2| + |y_1 - y_2|$$

This constraint defines a rectangular distance between two 2D vertex positions. We use it to formulate an algebraic minimisation constraint C_3 for the distance $dist$ on each axis:

$$(|x_1 - x_2| > dist) \vee (|y_1 - y_2| > dist)$$

If constraint C_3 holds for two positions P_1 and P_2, then the distance between both positions is at least $dist$. Since Choco provides no option to formulate an absolute value in a constraint and we reformulate $(|x_1 - x_2| > dist)$ as:

$$((x_1 - x_2) < -dist) \vee ((x_1 - x_2) > dist)$$

and the values of the variables are adjusted so that the constraints are again satisfied. The perturbation model has been used in interactive graphics systems like Sketchpad[17], ThingLab 1[18], Magritte[19], and Juno[20], and user interface construction systems such as Garnet [21].

[8] *http://www.choco-solver.net*

Aesthetic Criteria Constraint Formulation. The first aesthetic criterion we formulate as CSP is to distribute vertices uniformly, the second is to keep vertices conform with the user expectations.[9] Our implementation of the first criterion is an optimisation that tries to spread the graph as much as possible over the available screen. The Choco system offers the possibility to search for a general solution and thereby to maximise or minimise a certain variable. We use this to implement a node spreading behaviour (C_4). The variable denoting the objective value is the total distance between all vertices that are not related to each other (figure 3(b)). Our implementation of the second criterion corresponds to avoiding the following behaviour: If the user changes a relation, the positions of the vertices also change to the optimal positions for all remaining vertices, whereby all inactive relations are closed. In conjunction with the uniform distribution constraint C_4, this can lead to a completely new graph layout. Since a considerable new graph layout contradicts the aesthetic criterion to keep vertices conform with user expectations (small changes are needed to avoid human cognition problems), we try to implement a behaviour that leaves as many vertex positions as possible untouched while optimising the next graphical layout. The idea behind that is to rank the vertex positions according to the importance for the user. The measurement we use for this is the time delay of the node clicks the user performs during navigation. Roughly speaking, all the vertices that have been clicked recently keep their position in the new graph layout, all the vertices that have not been clicked recently release their positions more easily (C_5).

3.2 Limits of the Constraint-based Programming Method

Although we can avoid inconsistent layout constraints ($C_{1...5}$ are consistent), we cannot exclude cases where the CSP returns no result. Limits of the CSP include that a solution, which satisfies all existing constraints, does not exist. In our case this happens for example, if the number of vertices is to large to fit on the available space. To avoid empty screens, we automatically reduce the amount of displayed vertices thereby reducing the amount of constraints to satisfy. Whenever the automatic layout algorithm is not able to find a solution to satisfy all the constraints, the amount of active vertices will be reduced to only those vertices with an active input or output relation to the current focus vertex. Due to this reduction, the inconsistencies will be iteratively eliminated, until a solution layout can be found. We illustrate this procedure in figure 4, adding three artificial relations Test1, Test2, and Test3, to our match example. In Step1, showing the match focus instance and three active relations, all additional test relations are inactive and the CSP solution is available. In Step2, the focus changes to the date instance *08.06.1958* and thereby activates the relation Test1 between the date and the result instance $Ereignis \rightarrow (incidence : redcard)$. The activation of Test1 leads to a new C_2-constraint of the distance between the date and the result vertex. The CSP solution correctly forces the connected vertices to stay closer together. In STEP3, the user clicks on the *Ergebnis* relation, which

[9] C_2 already avoids edge crossings and keeps edge lengths uniform.

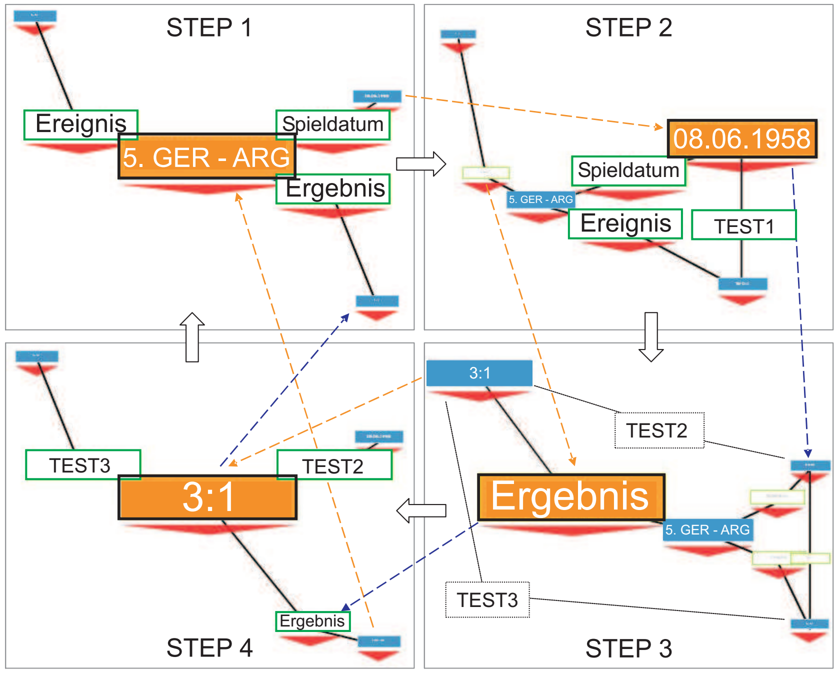


Fig. 4. Handling of inconsistent constraints.

activates the instance *3:1*. The invisible inactive relations TEST2, TEST3 are shown for illustration. The new C_2-constraints now force the TEST2, TEST3 related vertices to stay closer together. STEP4, clicking on the result instance *3:1*, results in an unsolvable CSP, because not all activated TEST1, TEST2, TEST3 relations can be placed in the *3:1* instance ring at the same time, without violating the central constraints $C_{1,2,3}$. On this account the layout algorithm is not able to find a solution to satisfy the central constraints of the six active relations and therefore the number of active relations/active nodes is reduced. As can be seen, only vertices with a direct connection to the current focus stay active and remain visible. This error recovery strategy resembles C_5 and turns out to be very robust.

4 Integration into a Distributed Dialogue System

A reaction and presentation component [22] is accountable for dialogue reaction and presentations in terms of the described semantic navigation structures. The added graph presentation planning capabilities includes (1) summarising multimodal result and finding an appropriate mapping toward a lower-level visual object and its attributes which we model in the interaction ontology, (2) finding out visual pattern interrelationships, (3) automating the visualisation of useful multimodal information which complements NLP generation output, and (4) provide consecutive information displays communicated from the server to the client. The semantic graph visualisation component is embedded into the reaction and presentation module. In SmartWeb, the graph-based user interface on the client is connected to the graph layout module that resists on the server. All data transfer between server and client is organised by special XML structures transmitted over socket connections in both directions. We extended this XML structure by an new *dynamic graph* environment, for the graph structure data to be exchanged, the graph node layout positions, and the user interactions. The data flow is shown in figure 5.

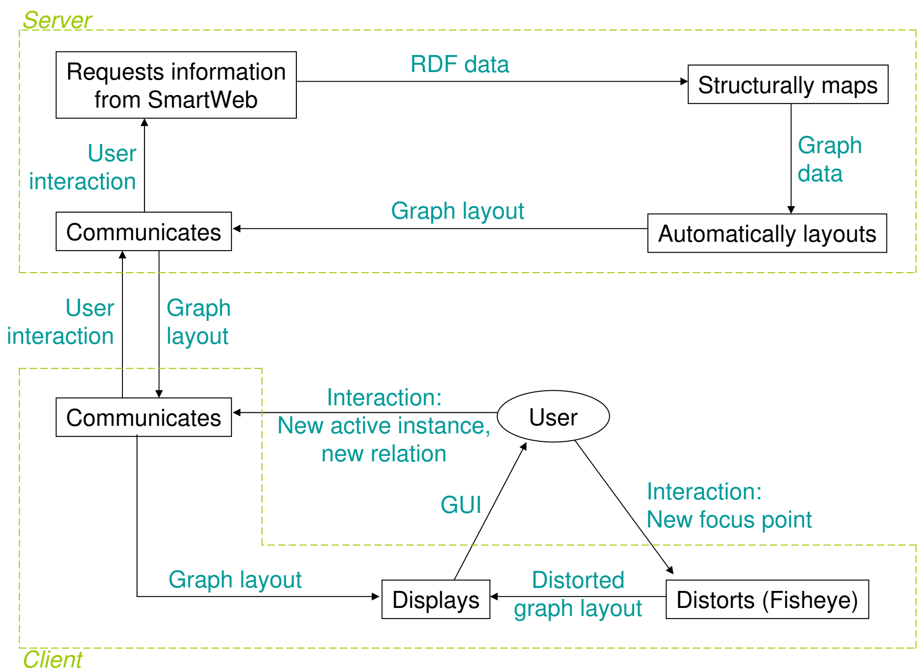


Fig. 5. Semantic graph visualisation data flow. Graph layouts for arbitrary RDF graph data are calculated on the server. On the client changing an active instance is communicated to the server.

The graph structure and layout are sent from the server to the client to be displayed on the handheld. If the user exchanges an ontology instance by choosing a new active instance in the focus point menu with new relations, a new football match instance for example, the interaction is sent back to the server to get a new layout. The possible user interaction, to change the focus point, is handled locally on the handheld client and needs no communication with the server.

5 Related Work

In the mobile question answering (QA) context we put coherence between consecutive questions and answer turns, as well as coherence between different answer views, such as the main GUI and navigations in focus. Work on representation recall ability and affordance mapping [23], automatic generation of personalised multimedia and hypermedia presentations for desktops [24], multimodal fission and media design for symmetric multimodality for the mobile travel companion Smartkom [25] heavily influenced our current design. DFKI's RDFSViz[10] provides a visualisation service for RDF data and uses the Graphviz[11] graph visualisation software to visualise ontology instances. Navigation is supported by retrieving relationships to other instances of the ontology. Desktop-based ontology navigation tools such as Ontoviz plug-in[12] for Protege support the retrieval of term definitions, or the drawing of complete ontology trees. Displaying RDF data in a user-friendly manner is a problem addressed by various types of applications using different representation paradigms [26]. At least the following types can be identified: keyword search, e.g. Swoogle[13], faceted browsing [27], explicit queries, e.g. Sesame[14], and graph visualisations. IsaViz[15] represent RDF models as vertex-edge diagrams, explicitly showing their graph structure. More advanced navigation tools and other interaction capabilities are provided by MoSeNa [28] and Haystack [29]. Generally limited display resources are addressed in [30]. Our account for data transformations of basically non-spatial or non-numerical relational information into graph visualisations is straightforward for configurations, where the graph size is incremental but does not present too many concurrent nodes and constraints. A constraint-based method for user interfaces, the DeltaBlue Algorithm, has been discussed in [31]; we used a simpler method with satisfiable time performance for interactive graph display.

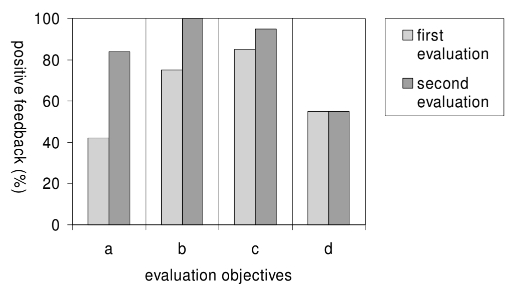
6 Conclusion and Future Work

We explored how to map highly structured RDF data as result structures in a QA scenario into a graph structure, and how the resulting graph can be visualised on a PDA, as an example of how to visualise Semantic Web data structures. We also discussed how multimodal interaction in dialogues can be established by additional graphical user interface capabilities—arranging RDF content through automatic layouting in a way that users better understand semantic relations of the contents by graph-based semantic navigation. The restrictions of the handheld device have also been taken into consideration, e.g., the small screen size, and we found a way to deal with the restricted computing power; the distributed

[10] http://www.dfki.uni-kl.de/frodo/RDFSViz/
[11] http://www.graphviz.org/
[12] http://protege.cim3.net/cgi-bin/wiki.pl?OntoViz/
[13] http://swoogle.umbc.edu/
[14] http://www.openrdf.org/
[15] http://www.w3.org/2001/11/IsaViz/

SmartWeb system turned out to be very useful for this purpose to compute automatic graph layouts on the server in order to solve the performance bottleneck on the handheld client. On the server, we used discretisation of the solution space to narrow down the search for a graph layout CSP to increase the reaction time of our interactive system. Using an automatic layout algorithm to arrange the vertices of the graph, our graph presentation system is able to deal with arbitrary RDF graph data. We finally proposed a way to handle situation where no solution exists to satisfy all layout constraints.

During the development of our constraint system we conducted two evaluation phases that involved 20 users to get feedback at an early stage (cf. [32]). We formulated four evaluation objectives: (a) *the possibilities to interact with the graph are easily to understand*, (b) *it is possible to extract information from the graph structure and the node labels*, (c) *the user gets aware of the difference between an instance and relation nodes*, (d) *the user realises the dependencies between related active instances whose labels appear.* A first evaluation showed that users had problems with changing layouts and extracting information from the distorted graph. By refining the fisheye distortion (from distance-based to topology-based distortion), text of non-focussed nodes could be displayed in a bigger font on the screen. Significantly, the formulation of the four aestetic criteria of section 3.1 also facilitates to understand the graph structure—as shows the second evaluation. 85% describe the graph interaction possibilities as easy to understand (after an initial demonstration), 95% easily understand the difference between instance nodes and relation nodes. The figures suggest users can get a more precise understanding of the presented information in its whole complexity. These feedbacks are useful sources of suggestions for the further improvement of our graph presentation system, and show additionally, that graph visualisations and interactions are generally welcomed alternatives for highly structured result data in QA scenarios. Further evaluations should focus on the questions



if users are able to reach and select the information they are most interested in. Extensions are editing and navigation functions via concurrent pen and voice to provide symmetric multimodal query fusion and query correction. We expect the concurrent use of pen and voice to reduce the initial learning phase toward intuitive use of multimodal mobile Semantic Web interfaces.

Acknowledgements

The research presented here is sponsored by the German Ministry of Research and Technology (BMBF) under grant 01IMD01A (SmartWeb). We thank our project partners, our reseach assistants, and the evaluators. The responsibility for this papers lies with the authors.

References

1. Wahlster, W.: SmartWeb: Mobile Applications of the Semantic Web. In Dadam, P., Reichert, M., eds.: GI Jahrestagung 2004, Springer (2004) 26–27
2. Sonntag, D., Engel, R., Herzog, G., Pfalzgraf, A., Pfleger, N., Romanelli, M., Reithinger, N.: Smartweb handheld - multimodal interaction with ontological knowledge bases and semantic web services. In Huang, T.S., Nijholt, A., Pantic, M., Pentland, A., eds.: Artifical Intelligence for Human Computing. Volume 4451 of Lecture Notes in Computer Science., Springer (2007) 272–295
3. Sonntag, D.: Interaction Design and Implementation for Multimodal Mobile Semantic Web Interfaces. In Smith, M.J., Salvendy, G., eds.: Human Interface and the Management of Information. Volume 4558 of LNCS., Springer (2007) 645–654
4. Wahlster, W., ed.: VERBMOBIL: Foundations of Speech-to-Speech Translation. Springer (2000)
5. Wahlster, W.: SmartKom: Symmetric Multimodality in an Adaptive and Reusable Dialogue Shell. In Krahl, R., Günther, D., eds.: Proc. of the Human Computer Interaction Status Conference 2003, Berlin, Germany, DLR (2003) 47–62
6. Reithinger, N., Fedeler, D., Kumar, A., Lauer, C., Pecourt, E., Romary, L.: MIAMM - A Multimodal Dialogue System Using Haptics. In van Kuppevelt, J., Dybkjaer, L., Bernsen, N.O., eds.: Advances in Natural Multimodal Dialogue Systems. Springer (2005)
7. Wahlster, W.: SmartKom: Foundations of Multimodal Dialogue Systems (Cognitive Technologies). Springer-Verlag New York, Inc., Secaucus, NJ, USA (2006)
8. Oviatt, S.: Ten myths of multimodal interaction. Communications of the ACM **42**(11) (1999) 74–81
9. Reithinger, N., Bergweiler, S., Engel, R., Herzog, G., Pfleger, N., Romanelli, M., Sonntag, D.: A Look Under the Hood Design and Development of the First SmartWeb System Demonstrator. In: Proceedings of 7th International Conference on Multimodal Interfaces (ICMI 2005), Trento, Italy (October 04-06 2005)
10. Oberle, D., Ankolekar, A., Hitzler, P., Cimiano, P., Sintek, M., Kiesel, M., Mougouie, B., Vembu, S., Baumann, S., Romanelli, M., Buitelaar, P., Engel, R., Sonntag, D., Reithinger, N., Loos, B., Porzel, R., Zorn, H.P., Micelli, V., Schmidt, C., Weiten, M., Burkhardt, F., Zhou, J.: DOLCE ergo SUMO: On foundational and domain models in SmartWeb Integrated Ontology (SWIntO). Journal of Web Semantics: Sci. Services Agents World Wide Web (2007)
11. Myers, D.G.: Psychology. Worth Publishers (2004)
12. Fensel, D., Hendler, J.A., Lieberman, H., Wahlster, W.: Spinning the Semantic Web: Bringing the World Wide Web to Its Full Potential. The MIT Press (2005)
13. Boselli, R., Paoli, F.D.: Semantic navigation through multiple topic ontologies. Semantic Web Applications and Perspectives, SWAP 2005 (2005)
14. Tsang, E.: Foundations of constraint satisfaction. Academic Press, New York (1993)

15. Di Battista, G., Eades, P., Tamassia, R., Tollis, I.G.: Graph Drawing: Algorithms for the Visualization of Graphs. Prentice Hall (1999)
16. Graf, W.: Constraint-based graphical layout of multimodal presentations. In: Advanced Visual Interfaces. (1992) 356–387
17. Sutherland, I.E.: Sketchpad: A man-machine graphical communication system. In: Spring Joint Computer Conference. (1963) 329–345
18. Borning, A.: The Programming Language Aspects of ThingLab, a Constraint-Oriented Simulation Laboratory. ACM Trans. Program. Lang. Syst. (1981) 353–387
19. Gosling, J.A.: Algebraic Constraints. PhD thesis, Carnegie-Mellon University (1983)
20. Nelson, G.: Juno, a constraint-based graphics system. In: SIGGRAPH. (1985) 235–243
21. Myers, B., et al.: The Garnet Toolkit Reference Manuals: Support for Highly-Interactive, Graphical User Interface in Lisp. Technical report, Carnegie Mellon University, Computer Science Department (1990)
22. Sonntag, D.: Towards combining finite-state, ontologies, and data driven approaches to dialogue management for multimodal question answering. In: Proceedings of the 5th Slovenian First International Language Technology Conference (IS-LTC). (2006)
23. Faraday, P., Sutcliffe, A.: Designing effective multimedia presentations. In: CHI '97: Proceedings of the SIGCHI conference on Human factors in computing systems, New York, NY, USA, ACM Press (1997) 272–278
24. Andre, E., Mueller, J., Rist, T.: WIP/PPP: automatic generation of personalized multimedia presentations. In: MULTIMEDIA '96: Proceedings of the fourth ACM international conference on Multimedia, New York, NY, USA, ACM Press (1996) 407–408
25. Wahlster, W., ed.: SmartKom: Foundations of Multimodal Dialogue Systems. Springer, Berlin (2006)
26. Pietriga, E., Bizer, C., Karger, D., Lee, R.: Fresnel: A browser-independent presentation vocabulary for RDF. In: International Semantic Web Conference. (2006) 158–171
27. Yee, K.P., Swearingen, K., Li, K., Hearst, M.: Faceted metadata for image search and browsing. In: CHI '03: Proceedings of the SIGCHI conference on Human Factors in Computing Systems, New York, NY, USA, ACM Press (2003) 401–408
28. Becker, J.and Brelage, C., Klose, K., Thygs, M.: Conceptual Modeling of Semantic Navigation Structures: The MoSeNa-Approach. In: Proceedings of the Fifth ACM International Workshop on Web Information and Data Management. (2003) 118–125
29. Quan, D., Huynh, D., Karger, D.R.: Haystack: A platform for authoring end user semantic web applications. In: International Semantic Web Conference. (2003) 738–753
30. Gerstmann, D.: Advanced visual interfaces for hierarchical structures. In: Human Computer Interaction. (March 2001)
31. Sannella, M., Maloney, J., Freeman-Benson, B.N., Borning, A.: Multi-way versus one-way constraints in user interfaces: Experience with the deltablue algorithm. Software - Practice and Experience **23**(5) (1993) 529–566
32. Heim, P.: Graph-Based Visualization of RDF Soccer Data and Interaction Possibilities on a Handheld. Diploma thesis, Computational Visualistics, University of Koblenz/Germany (2007)