



nature of the connections is dynamic meaning that they can change as the user changes the context of locally explored areas.

Building a user interface for browsing and exploring semantic space is a difficult task. Our goal was to hide the internal structure of RDF from the user and make the navigation process easy, enjoyable, rich and still meaningful. At the same time we were aware of the complexity of the data we were dealing with.

## RELATED WORK

Semantic Web data is modeled as a graph and thus many graph visualization and browsing techniques can be utilized to explore its content. However, it is not a purpose of this paper to analyze and discuss generic graph visualizations. It is important to point out that many of those visualization techniques do not scale to large datasets [1]. In our paper we are interested only in approaches focusing on semantic data.

One of the widely used techniques for exploring RDF data is faceted browsing [2]. It allows the users to find items based on more than one dimension. One of the problems is that faceted browsers use textual interfaces. Usually data is displayed either in a tabular (table) or linear (lists) format. Using such display techniques prevents the user from visualizing the space in a higher dimensional manner.

Disco [3] is a browser for navigating the Semantic Web as an unbound set of data sources. To start the exploration the user has to provide a resource URI. The interface displays information about the specified resource in a table containing property-value-source triples. The user can navigate between the resources following the hyperlinks which are included in the description of the current resource.

Open Link RDF Browser [4] allows exploration of the semantic space in a similar manner to Disco browser. Additionally, it provides several customized views (e.g. Yahoo Map, Timeline, Tag Cloud) that use data-specific interfaces to visualize and interact with underlying RDF triples. To use these special views the processed data has to contain certain information required by a specific component.

Protégé [5] is an ontology development tool. It allows visualizing edited ontology using node link representation. Such an approach focuses mainly on the structure of the underlying graph, including its concepts and relationships between them.

Some ontologies like UniProt [6] have too many entities and links to be easily visualized at once. In such cases exploratory techniques are necessary, enabling users to expand only the areas of interest. Paged Graph Visualization [7] takes this approach and allows the user to incrementally explore and visualize relevant RDF data. The Atom Interface follows the same approach. Instead of

showing the entire graph it allows the user to explore the space in a step-by-step manner.

Ontosphere 3D [8] is a rich 3D ontology visualization and editing tool implemented as a Protégé plugin. Concepts are rendered as spheres, instances as cubes, literals as cylinders and the relationships between entities are represented by arrowed lines. The user can manipulate the view through rotation, panning and zooming. Although the third dimension brings one more degree of freedom to the world of visualization it can also make the browsing activity more complex for the user. Additionally, it might be more difficult to understand the structure of the information displayed in the 3D environment.

All mentioned browsing techniques for semantic space do not permit one to navigate the data from different points at once. In our solution we follow an Interlinked Multi-Focus paradigm as we believe that it can be especially valuable for exploration of highly interlinked Semantic Web data.

## DESCRIPTION OF THE INTERFACE

For simplicity we break the description of our design into two elementary pieces. We start with a single unit called Atom and later on show how multiple instances of these units work together on one screen.

The Atom Interface presents users with a 2D surface containing several elements corresponding to single nodes from our initial graph (or graphs). Every element can be moved around the screen in a free manner and explored independently. Furthermore, a user is able to see connections between separately explored parts of a graph.

### Atom

Let's focus on a single element. As we pointed in the previous section it represents a single node from a graph. We call this node focus and from that point we start the exploration process.

### *Compact Radial Layout*

For the purpose of layout we treat the graph as a tree rooted at the focus node. Every level of exploration is represented as a ring (shell) containing all the child nodes belonging to the selected node on the preceding shell (or to the root node). The newly created shell always becomes the outermost one embracing all the rings representing the preceding levels. The focus node is always in the center. Child nodes within a shell are represented as circles with text labels. Circles corresponding to parent nodes have thin lines behind them (see Figure 2). The sizes of circles are in direct relationship with the number of children they contain. The circles are packed close to each other in a compact fashion and displayed around the parent node highlighting the 'belonging' relationship.

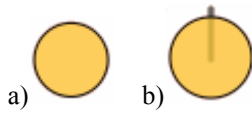


Figure 2. a) end node, b) parent node.

The circular organization of the information (shells and nodes) ensures that the distances between subsequent nodes are shorter and that the targets have more optimal shapes in terms of aiming and selecting.

**Behavior**

The interface was designed to behave similarly to a context menu. By right clicking on the focus node the first level expands, subsequent nodes expand if they are left clicked.

This way the user builds the exploration path incrementally uncollapsing (collapsing) the shells as they become relevant (irrelevant).

As the user moves away from the focus node each level increases in size. To enhance the scalability and make better use of the screen real estate our interface allows a user to zoom in/out (ctrl+mouse roll).

The user can perform circular scrolling of the currently expanded level arranging the nodes in a convenient way. A regular expanded tree has the tendency to look similar to a 'slice of pizza' thus allowing the user to pan the tree near one of the corners of the screen.

For example of a browsing sequence see Figure 3.

**Animation**

An important part of the interface is the animation. By clicking on a node (parent) its children expand in a fan-like manner capturing the user's attention on the newly revealed level. The animation of the last level produces so-called pop-out effect.

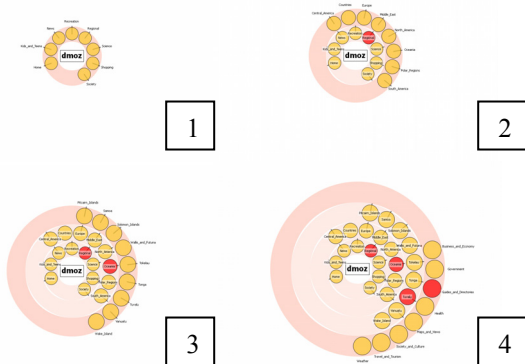


Figure 3. Browsing Sequence.

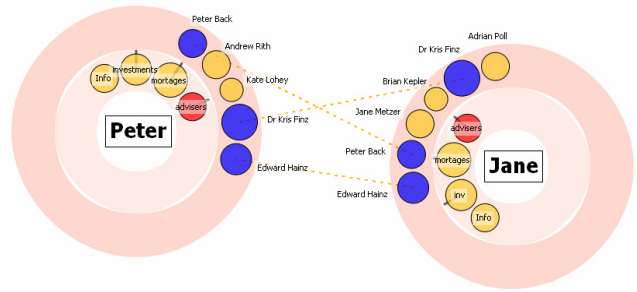


Figure 4. Two interconnected atoms (molecule).

**Preservation of context**

During the exploration process new shells and nodes are revealed. However, the context is fully preserved and the user can see the entire path leading from the central to the last selected node. This helps with localizing the information and understanding its origin (see Figure 2).

**Limitations**

Shells closer to the central node are spatially smaller and provide less space than the subsequent shells. If the graph has too many nodes on the first level one can consider changing the data structure itself by applying some clustering algorithms. The problem can be also mitigated leaving the data unchanged and using such techniques as fish-eye where only those nodes under focus are enlarged.

Deep hierarchies require many shells to be created. The screen space becomes a limitation here. The problem can be overcome by using a zooming technique (which is implemented in our prototype). Sometimes the user doesn't need to see the full path from the center to the certain node and is more interested in exploring a tree further from that particular node. In this case the context isn't important and the node of interest can become a new root node meaning that the inner shells can be discarded.

**Molecule**

A single atom component allows exploring and browsing a graph starting in one certain node. However, the interface is capable of displaying many atoms at the same time. All of them can be bound to different nodes of the same graph (or different graphs). The user is able to interact with the atoms independently browsing and exploring different parts of semantic space simultaneously.

The nodes from various atoms can be interconnected (see Figure 4).

Visualizing more than one atom simultaneously gives an insight into the data from many different perspectives. Connections between the nodes of separate atoms helps to relate the data, conclude, make decisions, understand, and grasp more details. This missed feature appears to be very crucial and useful when dealing with Semantic Web data which can be highly interconnected.

## Metaphor

Imagine that we have a huge graph which is our semantic space. We put atoms in arbitrary locations in the graph to explore and browse areas around those locations independently and simultaneously. The atoms are our spots of insight into the semantic space representing different perspective of view at the data. Furthermore, we can see the connections between those perspectives (atoms). The connections can be either explicitly defined in the graph or inferred dynamically in the background while the user is building his sub-space of exploration.

## EXAMPLE APPLICATIONS

We implemented a prototype of the Atom Interface and applied it to two different types of datasets in different domains. The first example application demonstrates a simple scenario where only one atom is used to browse DMOZ [9] taxonomy. DMOZ is the largest, most comprehensive human-edited directory of the web. It is constructed and maintained by a vast, global community of volunteer editors. The second application uses multiple atoms to explore social network graph.

### DMOZ Browser

We used a tree of about 1800 nodes obtained by trimming the DMOZ RDF dump. Every category from the taxonomy was mapped to a node described by an appropriate label.

User starts with a root node representing the entire DMOZ taxonomy. Right click opens the initial shell containing all the categories on the first level of the hierarchy. As user clicks on subsequent nodes new shells are revealed and new sub-categories appear. This way user builds his exploration path in a step-by-step process focusing more and more on details.

The context of exploration is fully preserved – all selected nodes are marked with red color. User is able to refine his

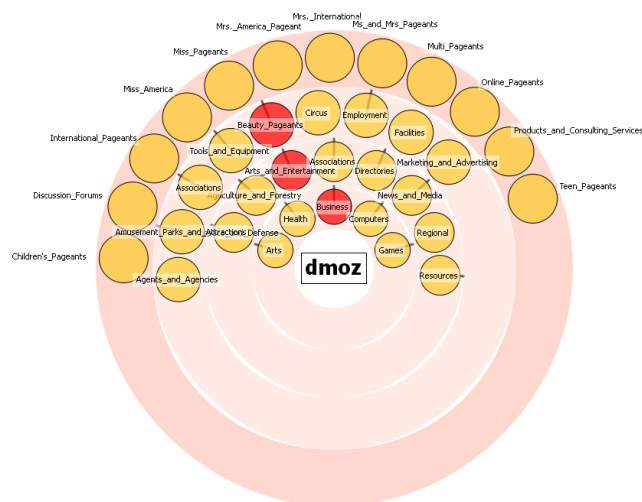


Figure 5. DMOZ Browser.

exploration path by clicking on any visible node. This way there is no need of going back step-by-step along the chosen path. A user can jump from one part of the taxonomy to the other. In such a case a number of shells can be collapsed before the new one is revealed. Since on every level apart from a selected node all the siblings are displayed it is easier for the user to modify and refine his choice (decision).

### Social network explorer

In this example application we wanted to demonstrate the use of multiple atoms. The data set represents people and their personal information: contact details, investments, loans, personal advisers, friends etc. For the purpose of this demo the data set was created manually and stored in RDF format.

Every person is represented by a single element on the screen. The personal information can be explored and browsed in the same manner as in the case of DMOZ Browser. As an example use we can explore local space of a particular person and see types of investments she has (e.g. properties, shares, funds). If we are interested we can explore these further and see, for example, all the child nodes of 'shares' node which are companies the person invested into.

By clicking on an end node (a node which is not a parent and thus doesn't have any children) a user can obtain more information regarding that node. For example, a user can select a node representing a company whose shares a particular person has. In such a case additional information about the company is displayed under the atom: number of shares, current share price, the annual turnover, and latest news from the company (see Figure 6). This additional information is attached to the atom. It means that if user drags the atom to a different location of the screen the information moves as well.

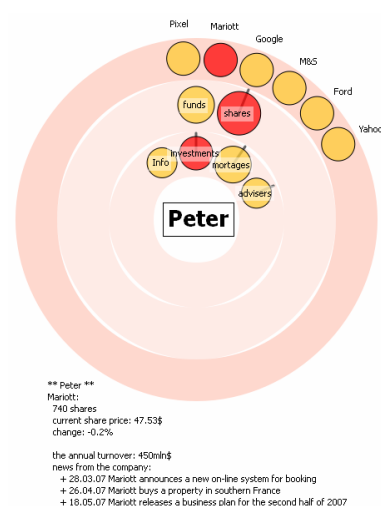
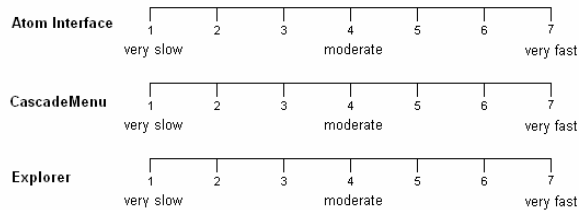


Figure 6. Atom with additional information.



How fast were you completing the tasks using interface X?



**Figure 9. Scale used for measuring subjective task completion time.**

Three types of tasks were used:

1. Retrieval tasks, e.g. : Find node “Spain”,
2. Go back to a previously found node e.g. : Return to the node named “Spain”,
3. Trace back tasks.

#### Procedure

Each evaluation session lasted approximately 1.2 hours and all participants worked on the same machine. The participants proceeded through: (a) introduction phase, (b) practice phase, (c) test phase, (d) post test questionnaire phase. During the introduction phase the authors demonstrated all the interfaces and their features, also describing the entire evaluation procedure to the user.

Practice phase let the subjects to experiment with the Atom Interface for five minutes to get familiar with its interaction behavior. All the participants expressed their familiarity with the Windows Explorer and the standard Cascade List Menu so no training was necessary. During this phase all the subjects were also presented with practice tasks. All three interfaces operated on test data which was not related to the data used in the experiment.

During the test phase every subject performed 3 different tasks for 5 concepts on each of the 3 considered levels starting with level 2 and moving towards level 4. We randomized the order in which the three interfaces were evaluated with respect to levels and subjects. After finishing the test phase the subject was asked to fill a questionnaire and give open-ended feedback about the 3 evaluated interfaces. The questionnaire was designed to measure user satisfaction in terms of specific subjective attitudes towards the interfaces. We used a 1 to7 scale to measure the user result in all situations (see Figure 9).

The user satisfaction questionnaire contains the following questions:

1. How fast were you completing the tasks using interface X?
2. How difficult was it to complete tasks using interface X?
3. Was interface X easy to use?
4. Was it enjoyable (fun) to use interface X?

5. Was it difficult to learn how to use Atom Interface?
6. Was Atom Interface intuitive?
7. How do you like Atom Interface?

The dependent variables were: the time to complete each task and subjective ratings on a 7-point scale.

#### Results and Discussion

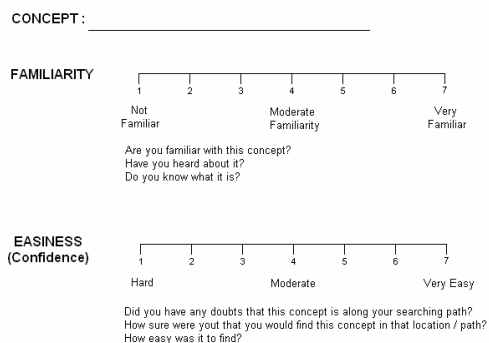
In this section we present all the tasks and corresponding hypotheses, followed by discussion of the obtained results.

#### Retrieval Tasks

Each subject was asked to find and retrieve 45 concepts distributed randomly in groups of 5 across the 3 interfaces, starting with level 2 concepts and continuing towards level 4 concepts. The performance is influenced by the subject’s familiarity with the concept to find [11] in terms of knowing a priori the location of the node in the tree. We feel that this factor is important to control when examining exploratory interfaces since clearly their purpose is to support search activities where the target may be even undefined. Many publications presenting exploratory interfaces pass over this problem and say nothing about the complexity of the tasks and how they were chosen.

For each question we ask the user to rank the familiarity of the concept before starting. After completing the task the user was asked to rank the easiness of the task on a 1 to 7 scale. For the purpose of this evaluation we did not use familiarity in evaluating the results. Easiness expresses the confidence the user had that the concept was where it was initially thought to be. We also provided questions to help the user better understand the definition of easiness:

- How easy was it to find?
- Did you have any doubts that this concept was along your searching path?
- How sure were you that you would find this concept in that location/path?



**Figure 10 Evaluation Questionnaires for Retrieval Tasks**



**Figure 11. Relative Performance in terms of easiness.**

These questionnaires (see Figure 10) gave us information on how easy the participants could determine the location of an item (concept). In this manner we were able to control the complexity of the tasks provided in our experiment and see whether they influence the overall performance of the interface.

We hypothesized that for easy tasks the Atom Interface should be more efficient than the considered interfaces in our experiment. According to Fitts' Law [12] the performance of pointing tasks is influenced by the distance and size of the target. In the case of the Atom Interface the radial layout provides optimal distances towards higher levels as opposed to list menu or explorer where the distance to the child nodes is increasing the further the child is in the list. Additionally box shapes that are used to display items in the Windows Explorer and the Cascade List Menu are more difficult to aim / target than circular shapes, used in the Atom Interface to display nodes. We consider easy tasks because the task completion time is mostly influenced by the interface and less influenced by the data set used.

**Results**

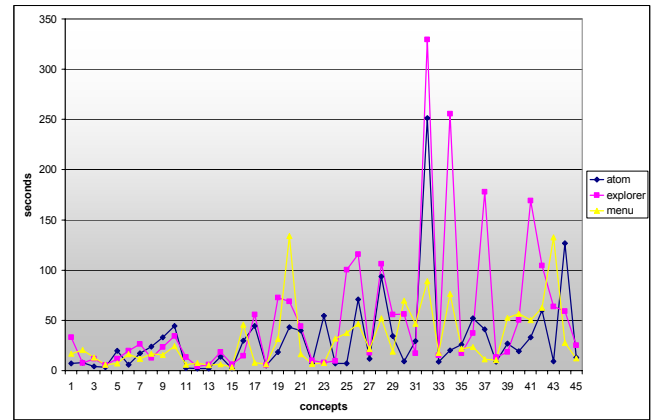
During the evaluation some of the users skipped certain questions due to their increased difficulty. These results were not considered in our analysis (e.g.: Subject would skip a question after 250 seconds). We measured performance in terms of level and difficulty (or easiness). The subjective easiness scores were normalized between 0 and 1 using the normal distribution for every user. For every interface we determined which were the easy and which the difficult question. We used the average normalized subjective score per question (from all the users who participated in answering this question for this interface) compared to two thresholds. The low threshold  $T_L$  was experimentally chosen to be 0.35 and the high threshold  $T_H$  was set to 0.65. Only questions falling outside the  $(T_L, T_H)$  interval were considered for analysis.

As can be seen in Figure 11 the Atom Interface was always better than Windows Explorer with an average time

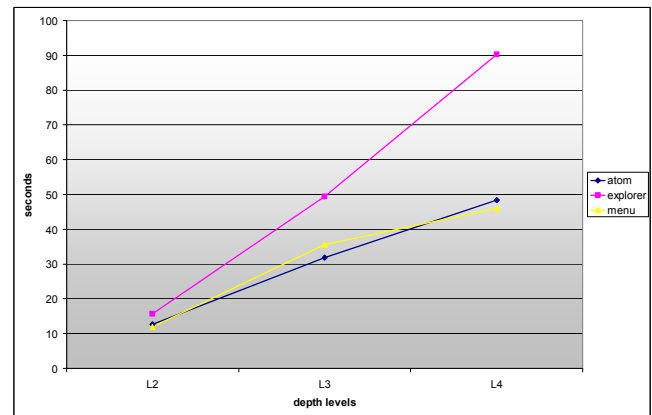
completion difference of 25 seconds for difficult tasks and 15 seconds for easy tasks. The Cascade List Menu interface surprisingly had an overall better performance for difficult tasks. We believe that this behavior is due to the fact that users perform less actions in menus than regular exploratory interfaces (e.g.: mouse clicks to show the next level – in menus the next level is displayed on mouse over), also animations slowed down the exploratory (Windows Explorer and Atom Interface) interfaces.

An interesting thing is that subjective perception of the task completion time (see Figure 18) always showed that the users perceived (were sure) that the Atom Interface was faster at achieving goals than the rest of the evaluated interfaces. We believe that this is due to the fact that people enjoyed using the Atom Interface more than the rest of the interfaces.

As can be seen in Figure 13, the average performance of the Atom Interface is always higher than Windows Explorer and comparable with the Cascade List Menu interface.



**Figure 12. Absolute Performance for Retrieval Tasks per concept basis.**



**Figure 13. Average Performance for Retrieval Tasks per depth level.**

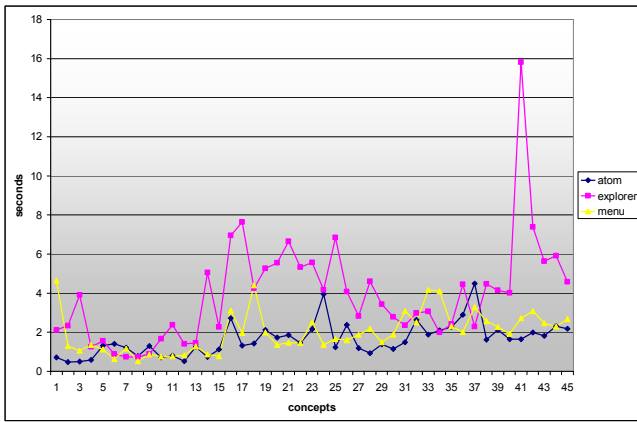


Figure 14. Absolute Performance for Trace Back Tasks per concept basis.

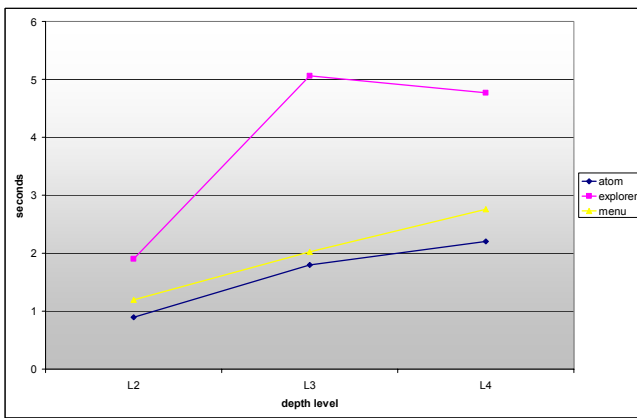


Figure 15. Average Performance for Trace Back Tasks per depth level.

### Trace Back Tasks

The subject was asked to click and select all parent nodes on the concept's path starting with the concept's parent going towards the root node of the hierarchy. This way we wanted to observe how fast the subject is able to track the local context.

Observation of the obtained results confirms that the Atom Interface was displaying an increased overall performance for this type of task (Figure 15). Again there is a consistent gap between Windows Explorer and the Atom Interface. We explain this through the fact that Windows Explorer like interfaces does not use the view port in an optimal way (the user has to scroll to get to the desired location e.g.: the parent node). The Cascade List Menu interface is comparable with the Atom Interface because the context was most of the times more condensed than in the case of Windows Explorer. The interface was slower than the Atom since the distance between parent nodes were greater on average and the shapes of the nodes are not easily aimable.

### Go back Tasks

The subject has to go back to the previously found node and select it again. The go back task started with a collapsed

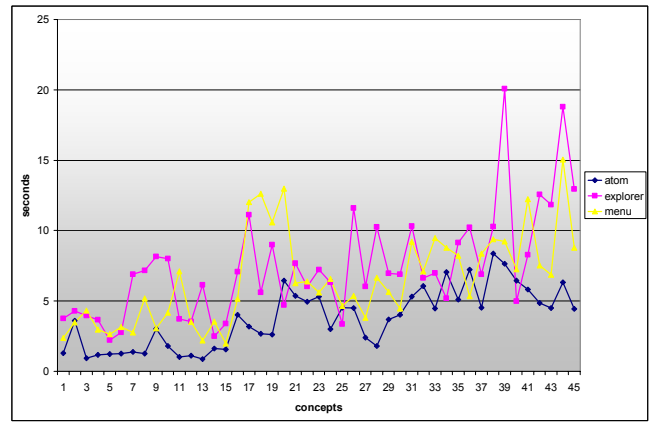


Figure 16. Absolute Performance for Go-Back tasks per concept basis.

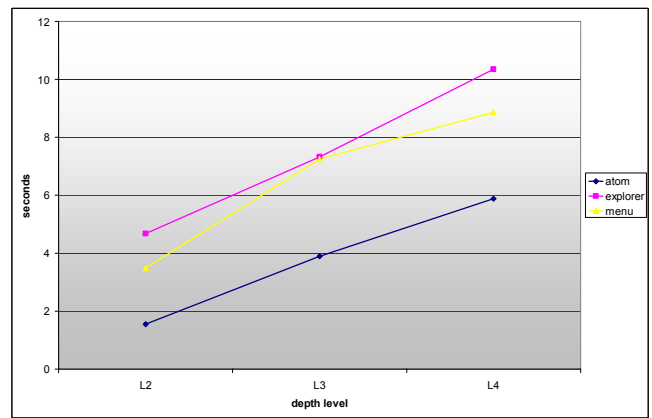


Figure 17. Average Performance for Go Back Tasks per depth level .

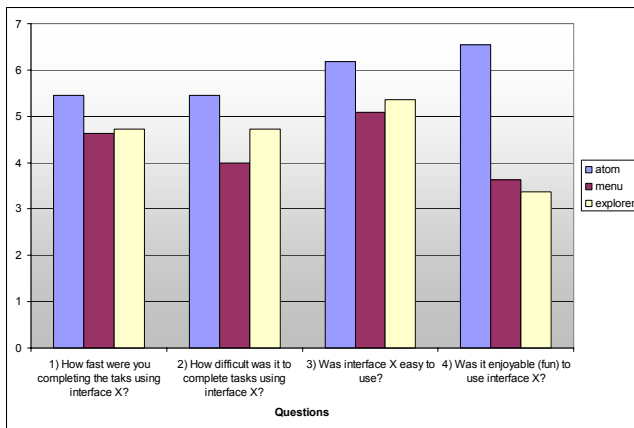
tree as the retrieval task. Also it was following immediately after the trace back task per concept basis.

We hypothesized that the Atom Interface will perform better in this type of task due to the fact that this is an easy task (the user already knows where the concept lies) the user could benefit more from the shorter distances between the nodes, the spatial arrangement of the nodes – patterns, and easily aimable shapes.

As can be seen in Figure 17 there is a big gap between the Atom Interface and the 2 other interfaces.

### Satisfaction

The last phase of the evaluation consisted of filling out a satisfaction questionnaire by the subjects. As mentioned by Hornbæk [13] sometimes we are interested not only in improving the objective performance but also in generating design advices on how to enrich users' experience of interaction. We hypothesized that because of the novel radial layout the interface would be more appealing and more engaging which would result in the users' perception and attitude toward the Atom Interface.



**Figure 18 Subjective user's satisfaction**

From the average obtained from the questionnaires we can clearly see that the Atom Interface was perceived to be the fastest the easiest and the most enjoyable among the evaluated interfaces. However not all the users found it intuitive, making common mistakes about the actions related to left and right click. The Atom Interface behaves both as a menu, and as an exploratory interface, thus right click will expand / collapse the tree (similar to context menus), and also keeping state consistent when moving the mouse across the interface just like a browser (in the Cascade List Menu, when the mouse was moved to a different sibling the previous node would automatically collapse and the new one expand automatically)

## DISCUSSION AND CONCLUSIONS

In this paper we presented a novel user interface – the Atom Interface – for browsing and exploring structured semantic web data. The interface uses introduced Interlinked Multi-Focus paradigm which allows exploring many areas of the information space simultaneously.

We also presented two example applications demonstrating the use of Atom Interface and conducted an initial evaluation of the compact radial layout used for information organization.

Although the users were given only 5 minutes to practice with the Atom Interface they were very efficient at performing most of the tasks (especially purely navigational ones).

Still for difficult tasks the interface is far better than regular exploratory interfaces (Windows Explorer), concluding that it also supports browsing in large information spaces where user needs can even be undefined. An additional benefit of the Atom Interface is the users' attitude towards the interface indicating that they find it enjoyable and engaging.

## FUTURE WORK

In the future work we would like to use interlinked data sets provided by Linking Open Data project. We are especially

interested in using the Atom Interface for visual integration of different information sources. We are also aware that there is a need for more elaborate evaluation and analysis focusing on semantic space exploration.

Furthermore we intend to augment the Atom Interface by providing more graphical cues increasing the adaptation potential of the interface.

Also different implementations for the interface will be considered (e.g. Flash implementation for web based navigation).

## ACKNOWLEDGMENTS

The Atom project was supported by the IDEAS project and eLITE project which is funded under the Enterprise Ireland Led Research Programme (Grant No. ILP/05/203).

## REFERENCES

1. Frasinca, F., Telea, A. and Houben, G.-J. Adapting graph visualization techniques for the visualization of RDF data. *Visualizing the Semantic Web*, 2006, 154–171.
2. Yee, K.-P., Swearingen, K., Li, K., Hearst, M. Faceted metadata for image search and browsing. *Proc. ACM Conf. Human Factors in Computing Systems*, CHI 2003.
3. Disco – Hyperdata Browser. <http://sites.wiwiw.fu-berlin.de/suhl/bizer/ng4j/disco/>.
4. OpenLink RDF Browser. <http://demo.openlinksw.com/DAV/JS/rdfbrowser/index.html>.
5. Noy, N.F., Sintek, M., Decker, S., Crubezy, M., Ferguson, R.W., Musen, M.A. Creating Semantic Web Contents with Protégé-2000. *IEEE Intelligent Systems*, 16 (2) 2001, 60-71.
6. Apweiler, R., Bairoch, A., Wu, C., Barker, W.C., Boeckmann, B., Ferro, S., Gasteiger, E., Huang, H., Lopez, R., Magrane, M., Matin, M.J., Natale, D.A., O'Donovan, C., Redaschi, N., and Yeh, L.S. UniProt: the Universal Protein knowledgebase, Jan 2004, *Nucleic Acids Res* 32, 115-119.
7. Deligiannidis, L., Kochut, K.J., and Sheth, A.P. RDF Data Exploration and Visualization. *CIKM 2007*, ACM, 39-46.
8. Ontosphere 3D. <http://ontosphere3d.sourceforge.net/userGuide.html>.
9. DMOZ Open Directory Project. <http://www.dmoz.org/index.html>.
10. Dumas, J. S. & Redish, J. C. (1999). A practical guide to usability testing, 2nd Ed.
11. John Lamping, Ramana Rao, Peter Pirolli, A Focus+Context Technique Based on Hyperbolic Geometry for Visualizing Large Hierarchies (1995), *Proc. ACM Conf. Human Factors in Computing Systems*, CHI.

12. Paul M. Fitts (1954). The information capacity of the human motor system in controlling the amplitude of movement. *Journal of Experimental Psychology*, volume 47, number 6, June 1954, 381-391.
13. Hornbæk, K., "Current Practice in Measuring Usability: Challenges to Usability Studies and Research", *International Journal of Human-Computer Studies*, vol. 64, issue 2, 79-102, 2006.
14. Card, S. K., MacKinlay, J. D. Schneiderman, B., (1999) Readings in Information Visualization: Using Vision to Think, *Morgan Kaufmann Publishers*.
15. Logan GD. The CODE theory of visual attention: an integration of space-based and object-based attention. *Psychol Rev.* 1996 Oct;103(4):603–649.
16. Atom Interface.  
<http://hci.deri.ie/~ksamp/atom.mas>.
17. Yee, K.-P., Fisher, D., Dhamija, R., Hearst, M. Animated Exploration of Graphs with Radial Layout, *IEEE Infovis 2001*.
18. Grosjean, J., Plaisant, C., Bederson., B. SpaceTree: Supporting Exploration in Large Node Link Tree, Design Evolution and Empirical Evaluation. In *Proc. IEEE Symposium on Information Visualization 2002*, 57-64.