Who is Who: On Visualizing Organizational Models in Collaborative Systems

Simone Kriglstein
SBA Research, Vienna, Austria
Email: SKriglstein@sba-research.at

Juergen Mangler and Stefanie Rinderle-Ma
University of Vienna, Faculty of Computer Science, Vienna, Austria
Email: juergen.mangler@univie.ac.at, stefanie.rinderle-ma@univie.ac.at

***Abstract*—Access control constitutes a key technology to fulfill security requirements in Collaborative Systems (CS) by specifying access rules/rights based on organizational models that capture the structure of the participating organizations. Organizational structures and consequently the describing organizational models can become complex comprising up to thousands of involved entities and relations. Further on, the management of organizational models is no longer only confined to specialists. Hence it becomes crucial to support the management of organizational information by adequate visualizations. Specifically, querying organizational models by, for example, finding out the number of associated actors to a specific role, has to be designed in a user-friendly way in order to avoid misunderstandings or even errors. As organizations are very likely to change over time, the user-friendly presentation of the related organizational models becomes even more important. This paper presents two novel visualization approaches called *OrbitFlower* and *OrbitList* for the analysis and management of organizational models in CS. Both approaches are evaluated with experts in regard to understandability of assignments of users to their roles/organizational units and visual properties.**

***Index Terms*—Collaborative systems, Visualization techniques, Organizational models**

I. Introduction

Over the last years an increasing number of studies has shown the importance of employing Collaborative Systems (CS) for communication and cooperation in organizations. However, for ”balancing the competing goals of collaboration and security” [1], adequate mechanisms to control access of users in CS are required. Access control can be found as a key technology to fulfill security requirements in CS [1]. Such access control approaches can be used in almost any field of applications to control users' access to any piece of information for collaborative tasks and the management of access control is no longer under control of a single administrator in the organization [2], [3].

Role-Based Access Control (RBAC) models can be seen as smallest common subset in existing systems and has become popular over the last years. The reason for the increasing interest is that RBAC models represent the organizational structure in a way that allows to reduce the administrative costs [3], [4], [5]. The basic assumption of RBAC models is that roles – that can reflect, e.g., responsibilities, activities, and functions within organizations – are more stable than users who can change easier over time [4], [6].

Organizational structures and consequently the describing organizational models can become complex comprising up to thousands of involved entities and relationships among these entities and groups [7]. As an example take the organizational structure of the Faculty of Computer Science at the University of Vienna as depicted in Figure 1 and extrapolate the size to large-scale enterprises.

Furthermore, the management of organizational models is no longer only confined to specialists. Organization models can be, for example, used to support managers in assembling teams for collaborative work, but can be also used by employees – who are sometimes non-specialists with the organizational structure – to detect other employees in their organization with similar tasks in order to share ideas, information or experiences.

The aforementioned reasons entail organizational models and the access rules defined on their basis to be hard to understand (particularly for non-specialists or users who are not familiar with the organizational structure). However, organizational models must be managed despite of this complexity in order to support organization's work, e.g., individually or in collaboration [7]. Therefore, visualization of organizational models will be central to the many CS in order to provide quick visual access to organizational structures that can help to reduce complexity and make information readily understandable. Users can build valuable knowledge while using a visual representation of the role-based access control structure. For example, visualization of dependencies between the roles in the organizations can support users to detect conflicts and to see the assignment of users to roles.

However, the development of user-friendly approaches for an effective usage of such access control approaches has received little attention [8]. In the last years, several visualization approaches for access control policies analysis have been published (see e.g., [4], [9], [10], [11], [12], [13], [14], [15], [16]). Although the arguments for usable access control approaches are well-known in the community, evaluation studies – conducted with users – of such visualization approaches are still difficult to find. To provide a user-friendly design of access control visualizations, it is important to consider potential users already during the development process in order to verify if the graphical representation meets users' needs. For this purpose, well-known techniques from the human-computer interaction area can be applied which in general have become more and more popular for the evaluation of

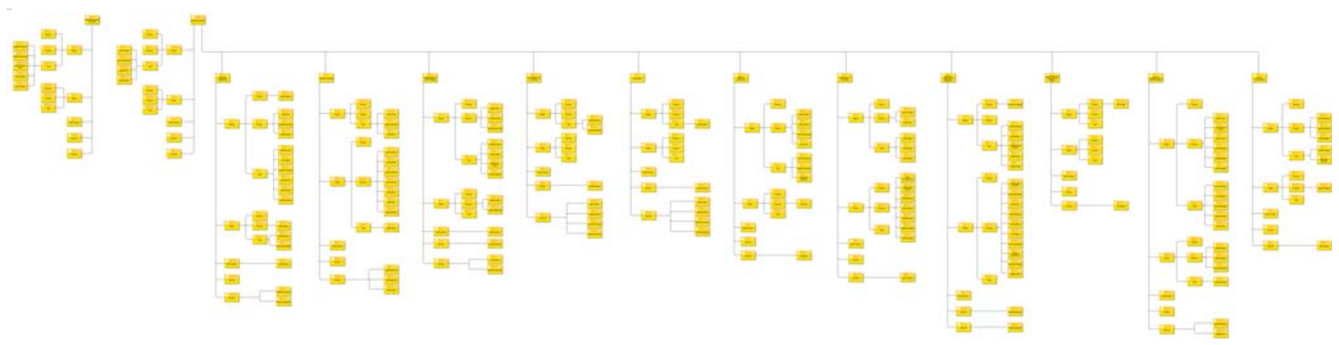

Fig. 1. Organizational structure of the Faculty of Computer Science at the University of Vienna (modeled as organigram by using ArisExpress, `www.ariscommunity.com/aris-express`).

visualization approaches over the last years (e.g., see [17], [18]).

The choice which layout should be used often depends on users' preferences that do not necessarily support task performance optimally. Therefore we want to integrate different layout approaches to enrich the web interface of the SPRINT[1] Enactment Engine [19] to provide an optimal support for users in consideration of their preferences and needs. We developed two visualization approaches – called *OrbitFlower* and *OrbitList* – to present the role-based access control structure. Both approaches visualize the dependencies between roles, organizational units, and users and consider the following different types of connections: *role-organizational unit* relation, *role-role* relation, *organizational unit-organizational unit* relation, *role-user* relation, and *organizational unit-user* relation. Since such organizational models can contain large numbers of relationships between roles, users, and organizational units, it is necessary that our design solutions help to avoid a cluttered display. Secondly, the different types of relationships have to be visually distinguishable from each other. Based on our intention to develop usable approaches for the analysis and management of organizational models, we consider potential users early in the development process in order to get early feedback to enable improvements of our approaches.

In this paper we present the design and implementation of both visualization approaches and an evaluation study in order to compare both approaches and to find out if the design ideas are understandable for users. The remainder of this paper is structured as follows. Section 2 presents the visualization designs of both approaches and the implementation is shown in Section 3. In Section 4, the evaluation study and results are presented. Observations based on the findings of the evaluation study are discussed in Section 5. In Section 6, related work is presented. Finally, the paper is concluded and gives an outlook on future work.

[1]In the SPRINT project, we are developing a security component and enactment engine on top of our adaptive process engine CPEE (see SPRINT project homepage: www.cs.univie.ac.at/project/SPRINT and CPEE project homepage: www.cpee.org)

II. Visualization Design

In this section we present the design of our two approaches called OrbitFlower and OrbitList (see Figure 2) to present relationships between organizational units, roles and users.

For the visualization of relationships between objects, there exist two possibilities (see e.g. the taxonomy of Shneiderman in [20]): tree and network. Network and tree visualizations consist of a set of nodes which are connected by a set of edges and often called as node-link representations. Tree visualizations are used to present hierarchical structure (or also called tree structure) and examples are Hyperbolic tree [21], Treemap [22], and SpaceTree [23]. However, sometimes the visualization of hierarchical relationships (*child-paren*t relationship) are not enough. Network visualizations are node-link representations that also link nodes to an arbitrary number of other nodes independent from their hierarchical relationships. In our case, our approaches are based on an extended RBAC model (see Figure 4). Therefore, a visualization of hierarchical structure is not sufficient, because the ternary relation `has_relation` in the center, describes that users have a particular role in a particular organizational unit. This includes that a user can have different roles in different organizational units, which actually was quite common in our datasets.

For the OrbitList approach a rectangular layout is used whereas the OrbitFlower approach is based on a circular layout. In a circular layout all nodes are located on the perimeter of a circle [24]. Connections between nodes are usually presented within the circle. This form of layout is often used to present networks and systems management diagrams [24]. A rectangular layout arranges nodes parallel to the x- and y-axes. Both layout designs have their strength and weaknesses. There exist several studies about various layouts for node-link representations depending on different applications (e.g., [25], [26], [27], [28]). For example, one of the major advantages of a circular layout is that the compact shape allows to handle large number of nodes. In contrast, studies (e.g., [25], [26], [28]) show that users are generally faster to complete tasks with a rectangular layout. Reasons could be that it is easier to read for the users if the nodes are arranged from left to

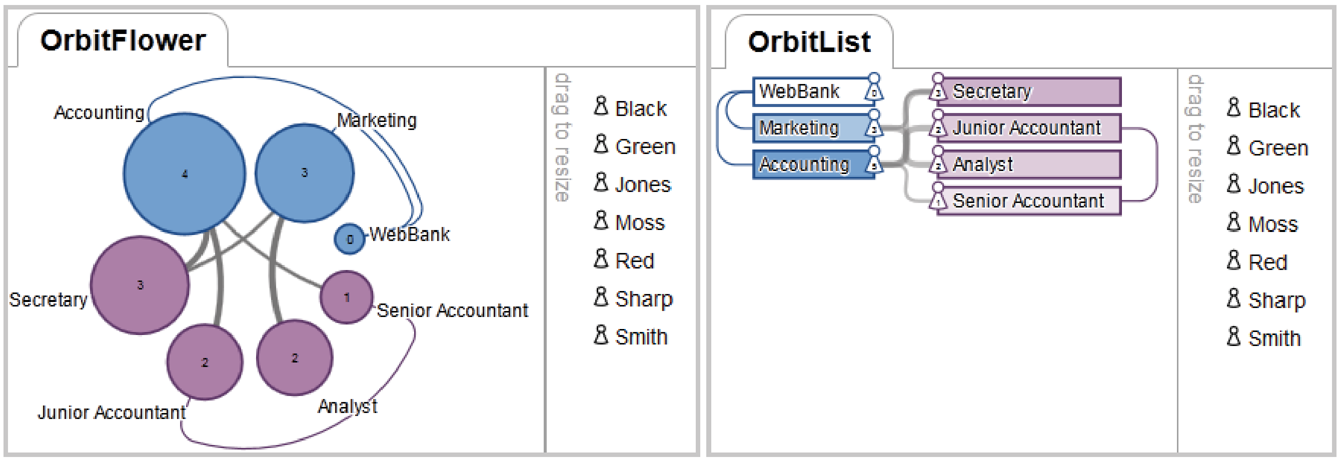

Fig. 2. The left side shows an example for the OrbitFlower and the right side presents the same dataset with the OrbitList.

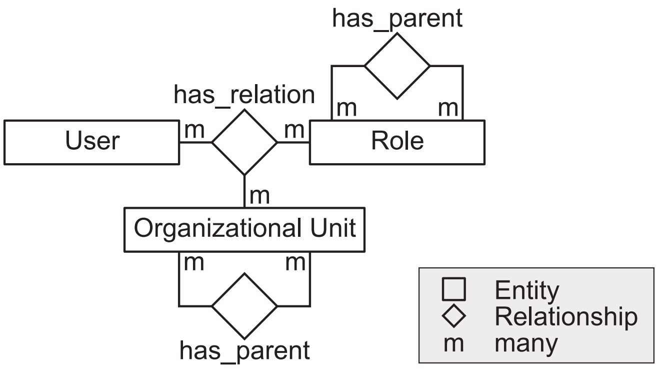

Fig. 4. Data model.

right or from top to bottom and therefore the clarity of the hierarchical structure is better than with a circular layout.

A. Basic Structure

Both, OrbitFlower and OrbitList visualize roles and organizational units as nodes. The nodes for roles and for the organizational units are defined as own node types. The following relationship types between the nodes are represented as lines:

- *role-role* relation and *organizational unit-organizational unit* relation: hierarchical structure of roles and organizational units.
- *role-organizational unit* relation: relationships between roles and organizational units.

In contrast to other approaches (e.g., [10], [12], [29]) which also represent users as nodes in the visualization, users are presented in alphabetically order as list in another view. This design decision helps to reduce the number of the visualized links between the nodes. Furthermore, the relationships between users and roles or organizational units can change often in organizations (e.g., users can leave the organization or their responsibilities change). Therefore, the representation of users in an own view enables a stable structure and helps to keep the abstract structural information in mind that is formed about the organizational structure over time. Brushing and linking technique is used to highlight the *role-user* relation and *organizational unit-user* relation that represent relationships between users and their assigned roles and organizational units. For the representation of users for a specific role or organizational unit, there exist two options:

1) if the mouse moves over an organizational unit or role, then the corresponding users are highlighted in the list or
2) if an organizational unit or role is selected via mouse click, then only the corresponding users are listed.

Figure 3 shows examples of the different options for the OrbitList (for the OrbitFlower it is realized in the same way). The arrangement of nodes and links is different in both layout approaches:

- **OrbitFlower:** The nodes are arranged on the perimeter of a circle and they are grouped depending on their node type (see left side in Figure 2). This arrangement allows users to see the roles and the organizational units at first glance and makes the distribution between organizational units and roles visible. The *role-organizational unit* relations are visualized within the circle and the relationships to present the hierarchy of the organizational units and roles are visualized outside of the circle.
- **OrbitList:** The nodes for organizational units are visualized on the left side and the nodes of roles on the right side of the visualization. Depending on their node type, they are listed among a line parallel to the y-axis (see right side in Figure 2) to allow for a better distinction between roles and organizational units. The *role-organizational unit* relationships are visualized in regard to minimize the length of lines.

For both layout approaches, the nodes of the same node type are arranged in a way to reduce edge crossings for the *role-role* relations and *organizational unit-organizational unit* relations.

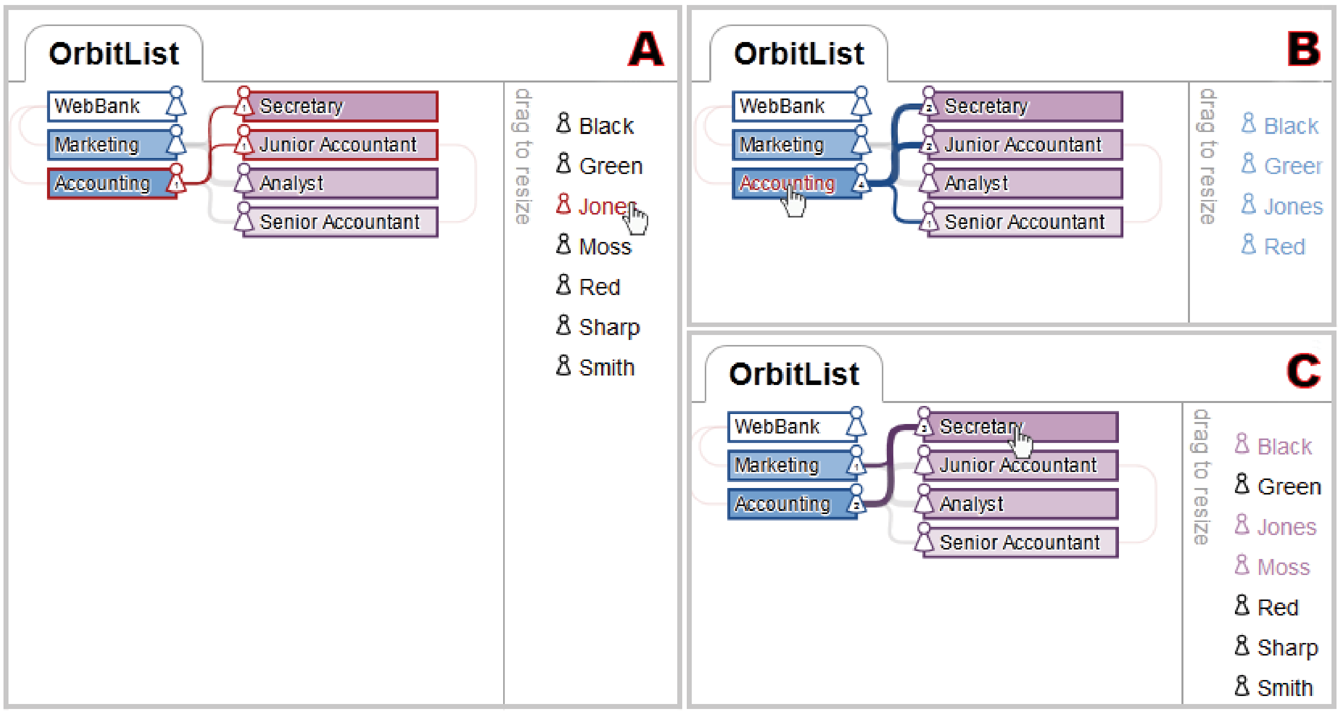

Fig. 3. Three examples how the connections between an organizational unit, role and their users are visualized in the OrbitList (for the OrbitFlower it is implemented in the same way): (A) shows an example if a single user is selected, (B) presents an example if the user clicks on an organizational unit, and (C) shows the case when a role is selected.

B. Visual Properties

For a better distinction between organizational units and roles, two different colors are used. The color blue is used for organizational unit nodes and their *organizational unit-organizational unit* relations and the color purple presents role nodes and their *role-role* relations. In addition to the color-coding, spatial separation was used in both layout approaches to make the *role-organizational unit* relations and the hierarchical relations better distinguishable. The lines for *role-organizational unit* relations are curved and are drawn with alpha-blending to minimize obscuration of other curves in both layout approaches. Furthermore, the thickness of the lines for *role-organizational unit* relations corresponds to the number of assigned users. Following further visual properties are used:

- **OrbitFlower:** Nodes (organizational units and roles) are visualized as circles and the size of the circle depends on the number of the assigned users to this node. In addition to the varied sizes of circles, the label in the center of the circle represents the number of assigned users (left side of Figure 5).
- **OrbitList:** Nodes (organizational units and roles) are visualized as rectangles and the intensity of the color for each rectangle (c.f. Figure 2) depends on the number of users assigned to this node. For the representation of the label of the number of assigned users, an own user icon is used (right side of Figure 5).

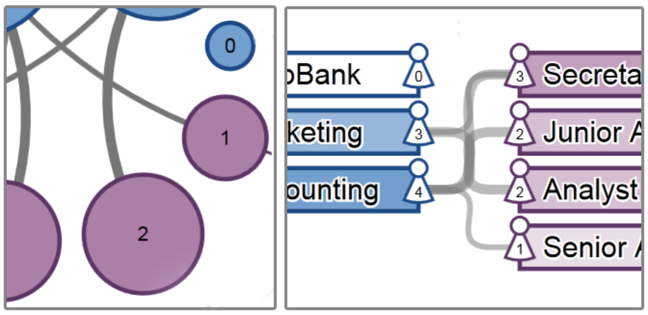

Fig. 5. Left (OrbitFlower): example for the different size of circles in combination of the labels that represent the number of the assigned users. Right (OrbitList): the user icon are used to highlight the number of assigned users.

III. Implementation

For the implementation of both layouts in order to enrich the web interface of the SPRINT project, each approach is realized as a component that takes XML (see Listing 1) as input, and Ruby is used to returns a piece of HTML5 with inline Scalable Vector Graphics (SVG), JavaScript and CSS. Figure 6 gives an overview of the interplay between the different parts.

List. 1. XML realization of the data model (see Figure 4) for the example that are visualized in Figure 2 and Figure 3.

```
<?xml version="1.0"?>
<organisation xmlns="http://cpee.org/ns/organisation
      /1.0">
    <units>
        <unit id="WebBank"/>
        <unit id=”Marketing”>
            <parent>WebBank</parent>
        </unit>
        <unit id=”Accounting”>
            <parent>WebBank</parent>
        </unit>
    </units>
    <roles>
        <role id=”Secretary”/>
        <role id=”Junior Accountant”>
            <parent>Senior Accountant</parent>
        </role>
        <role id=”Analyst”/>
        <role id=”Senior Accountant”/>
    </roles>
    <subjects>
        <subject id=”Black”>
            <relation unit=”Accounting” role=”Secretary”/>
        </subject>
        <subject id=”Jones”>
            <relation unit=”Accounting” role=”Junior Accountant”/>
            <relation unit=”Accounting” role=”Secretary”/>
        </subject>
        <subject id=”Red”>
            <relation unit=”Accounting” role=”Junior Accountant”/>
        </subject>
        <subject id=”Green”>
            <relation unit=”Accounting” role=”Senior Accountant”/>
        </subject>
        <subject id=”Smith”>
            <relation unit=”Marketing” role=”Analyst”/>
        </subject>
        <subject id=”Sharp”>
            <relation unit=”Marketing” role=”Analyst”/>
        </subject>
        <subject id=”Moss”>
            <relation unit=”Marketing” role=”Secretary”/>
        </subject>
    </subjects>
</organisation>
```

IV. Evaluation

The motivation of the evaluation study in the early development phase was to detect misinterpretations or unclear elements and to get early feedback about the fundamental design ideas of both visualization approaches. Hence, the comparison between the design of OrbitFlower and OrbitList help us to find out advantages and drawbacks of the two layouts. The results of the evaluation will indicate possible improvements of the design which should be considered for the redesign. For these purposes, the main questions of the evaluation study are:

1) Are the visual representation and the spatial separation helpful to distinguish between organizational units, roles, and users?
2) Are the visual representation and the spatial separation helpful to distinguish between relationship types?
3) Are the assignments of users to their roles and their organizational units clear?
4) How useful are the different visual properties provided by OrbitFlower and OrbitList?

A. Methodology

For the evaluation, we used the following methods: observations in combination with thinking aloud, and semi-structured interviews. These research methods are well-known methods from the human-computer interaction area and are often combined for the evaluation of information visualizations. Observations help to identify what users like or dislike of the design, but can alone easily lead to misinterpretations [30]. Therefore, it is often used in combination with thinking aloud to make users’ decisions knowable. This is especially helpful when users act in unexpected ways [31]. Interviews are often used to find out what users think about the design [30].

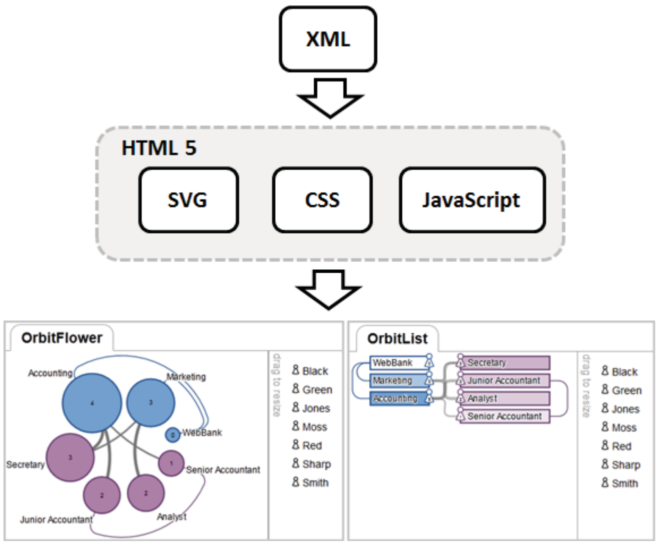

Fig. 6. Overview of the interplay between the different parts.

B. Procedure

Before the participants started with the analysis of both visualization approaches, the structure of the organizational model was introduced. After the introduction and a description about the purpose of the evaluation study, the participants started with the evaluation and they were asked to interact with the visualizations in order to fulfill different tasks. The focus of one set of tasks was the identification of organizational units, roles, and users as well as their different relationship types. Another set of tasks concentrated mainly on the different visual properties. While the participants interacted with the visualization approaches, they were observed and encouraged to think aloud. After they finished the tasks we asked them which visualization approach they preferred with regard to aesthetics and utility. Furthermore, they were asked about the strengths and weaknesses of both visualizations and to compare them with visualizations which they usually use to analyze the organizational structure.

C. Sample

OrbitFlower and OrbitList were tested with 13 experts. Seven from the 13 participants have high expertise in the field of business processes and have at least basic knowledge about information visualization. The remaining six participants are experts in the field of information visualization and have at least basic knowledge about business processes. The reason for using experts was to receive valuable qualitative feedback and therefore it was important that they were familiar with the concepts of information visualizations and the structure of organizational models. Testing sessions for each participant took about 40 minutes. As test case an organizational model was used that present the interconnections of organizational units, roles, and users from the Faculty of Computer Science at the University of Vienna (c.f. Figure 7 and Figure 8). This organizational model includes 13 organizational units, 8 roles, 117 users, 239 *role-organizational unit* relations, 6 *role-role* relations, 11 *organizational unit-organizational unit* relations, 239 *role-user* relations, and 239 *organizational unit-user* relations. Because of the data model, which requires that relations always occur between a user, a role and an organizational unit, the number of *role-organizational unit* relations equals the number of *role-user* and *organizational unit-user* relations.

D. Results

The findings are based on the qualitative analysis and are presented according to our research questions.

1) Representation of Organizational Units, Roles, and Users: All experts had no problems to identify the groups of organizational units, roles and users in both visualization approaches. Especially for the OrbitList, the spatial separation helped them to immediately see roles and organizational units. Although most of the participants found the layout of the OrbitFlower to be ”nice”, ”beautiful”, ”exciting” or ”cool” and it was ranked higher (11 from 13 participants) concerning aesthetics, nine participants stated that the structure and the distribution of roles and organizational units were better understandable in the OrbitList at the beginning. Four experts noted that the labels were difficult to read in the OrbitFlower, because they overlapped with the hierarchical relationships and that it was easier for them to scan the labels in the OrbitList. Furthermore, they needed more time to get an orientation in the OrbitFlower, because they were more familiar with the list-representation of roles and organizational units. As advantage of the OrbitList it was mentioned that this layout design provides a clear starting point to analyze the organizational units and roles from top to bottom or from left to right. Furthermore, it was pointed out by two participants that the OrbitList was useful to make a rough estimate of how much nodes were organizational units and how much nodes were roles. These were also the reasons why most participants (12 from 13 participants) ranked the OrbitList higher concerning utility than the OrbitFlower. In this context it was often named that it depends on the tasks which they would like to solve. For example, OrbitFlower would be more useful to get a first overview about the organizational structure and dependencies, but the OrbitList would be better to answer specific questions. In comparison with node-link representation which they usually used (e.g., UML diagrams), they found that OrbitList and OrbitFlower provided a better overview especially for large organizational models. The participants found that especially the OrbitFlower would be better suitable for larger datasets. For example, one expert noted that the arrangement of nodes in the OrbitFlower would enable him/her to memorize the position of nodes easier (e.g., a 2 o'clock position) than a position of a node in the OrbitList in case scrolling would be necessary.

2) Representation of Relationships: In comparison with visualizations which they usually use, they noted as weakness that often no visual differentiation between the different relationship types exists. The identification of the different relationship types was for all participant no problem, because of their representation and their spatial separation. Only the semantic meaning of the *role-role* relations and the *organizational unit-organizational unit* relations was for some experts difficult to understand at the beginning. The problem was that the identification of the parent nodes and their children nodes was not clear enough and especially in the OrbitList, the representation of organizational units and roles as lists misled the experts to assume a hierarchical order. They noted that additional visual properties would be useful to provide a better differentiation (e.g., by using different colors) and to see hierarchical dependencies (e.g., to highlight also the parent node after one of its children nodes is selected). Furthermore, it was for several participants confusing that the hierarchical representation was only visualized in one direction, i.e., that only *parent-to-children* relations are shown. For the analysis, they noted it would be helpful to see the hierarchical dependencies in both directions (*parent-to-children* relations and *children-to-parent* relations). In general, most experts found the representation of the hierarchical structure better in the OrbitList than in the OrbitFlower. They stated that lines were very dominant in the OrbitFlower and it was more difficult to follow them because of edge crossings and because the edges were longer. Although the number of edge crossings was equal in both approaches, one experts noted that the advantage of the OrbitList was the orthogonal angles between the crossed edges and therefore it was easier to follow the lines between start and endpoint than in the OrbitFlower. The *role-organizational unit* relationships were for all experts clear. One participant suggested that it would be helpful to highlight also the corresponding hierarchical relationships if a *role-organizational unit* relation is highlighted. In contrast to the representation of the hierarchical dependencies, five experts found that the visualization of *role-organizational unit* relationships is better comprehensible with the OrbitFlower, because of the wider representation. It was also noted that the circular layout of the OrbitFlower provided a better overview of the flow of the *role-organizational unit* relationships as the OrbitList in case no highlighting for the connected lines was used.

3) Assignments of Users to their Roles and their Organizational Units: The representation of users in an own view to spatially separate them from their roles and organizational units was noted as a good solution in order to reduce the number of lines. They stated that other visualization approaches which presents the users as nodes and the assignments to

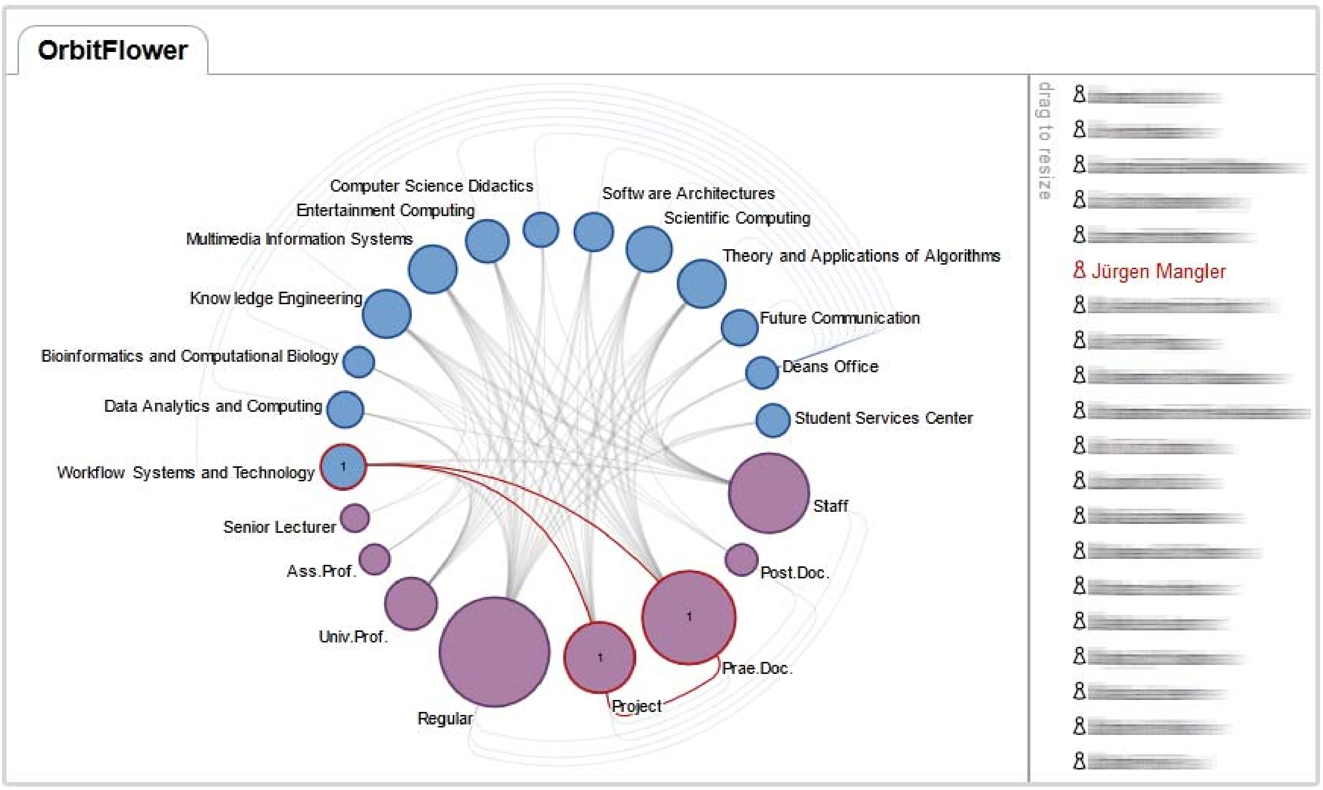

Fig. 7. This OrbitFlower example shows the assignment of the user "Juergen Mangler" to his organizational unit and role.

their roles and organizational units as lines, are often unclear because of the large amount of lines and nodes, especially for larger datasets. Furthermore, one expert pointed out that the splitting between users and roles/organizational units reflects the organizational structure and the assignments of their users better. The connection between the two views via linking and brushing was for all participants clear and well-received. Only the multi-assignments of users were for five experts (mainly visualization experts) difficult to understand, because they were confused that the number of assigned users for roles and their distribution to the organizational units and vice versa can be different. One expert noted it would be useful to connect only the child node via the *role-organizational unit* relations. For example, in Figure 7 this would mean to show only the connection between organizational unit "Workflow Systems and Technology" and the role "Prae. Doc.". Furthermore, they missed the number of the assigned users to their roles/organizational units in order to analyze the distribution of the hierarchical dependencies (to see e.g. in Figure 8 also the distribution of the assigned users for the roles "Staff", "Post. Doc.", and "Prae. Doc."). In general, they liked to interact with both approaches and no distinctions were observed between OrbitFlower and OrbitList.

4) Visual Properties: The usage of colors and the spatial separation of nodes and lines was useful to identify the different semantic groups. Only that the color used for highlighting also reflected the corresponding color for organizational units and roles was confusing at the beginning. It was noted that a uniform highlight color would be sufficient, because the spatial separation of the groups was clear enough. Furthermore, the meaning of the thickness of the lines was for all experts clear. The meaning of the different sizes of the circles was also immediately understandable for all participants. It was noted that the size made it easier to compare the nodes in regard to the number of assigned users than with the intensity of the color. The reason was that the intensity of the color was not dominant enough and misled five experts to make wrong assumptions (e.g., the intensity depended on the number of the children nodes). Four experts pointed out that nodes with the maximum value and minimum value were better comparable than nodes that were colored with similar intensity. Furthermore, most of the experts found the label for the number of the assigned users difficult to read, especially for multi-digit numbers in the OrbitList. The named reasons were the small font size and the user icon that was only a graphical element without additional benefit for the visualization. They would have preferred a simple representation (e.g., no icon or only a rectangle instead of the user icon). In comparison with the visualizations which they usually use, such as UML diagrams, they liked the usage of the different visual properties to provide additional information (e.g., to see the distribution of the assigned user with the thickness of the lines, or to be

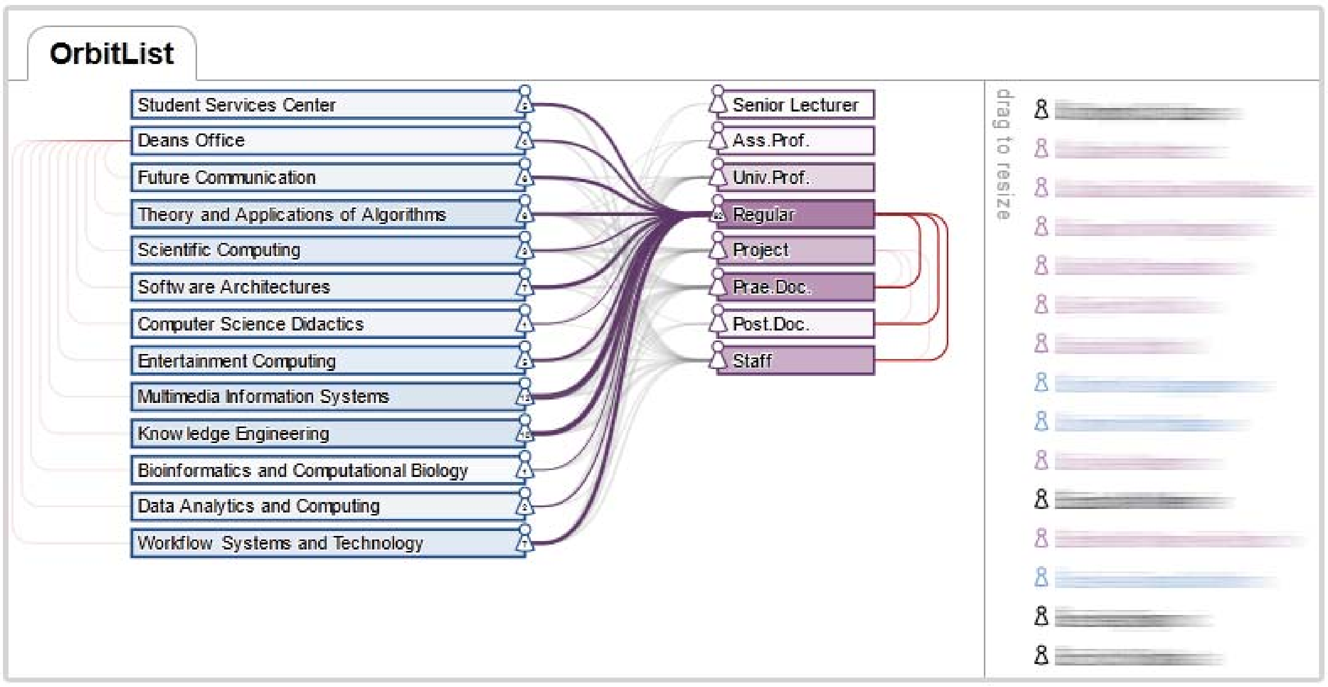

Fig. 8. The OrbitList example shows the assignment of users to the role "Regular".

able to compare the number of users by looking at the size of the circles).

V. Discussion

The organizational model builds the fundament for access control in CS, i.e., access as well as authorization constraints for the processes are defined on its basis. Access rules assign tasks to authorized users, e.g., *Role="Univ.Prof"* for task *Supervising thesis.* Authorization constraints might further constrain the access rules such as separation or binding of duties (SOD/BOD) [19], [32]. It is straightforward to argue that modeling organizational structures becomes easier and less error-prone if users can rely on a visualization they understand. Figure 1 depicts the organizational model as visualized in Figure 7 and Figure 8. The findings of the evaluation point out that the separation of users from roles and organizational units in the OrbitFlower and OrbitList is the reason that the organizational structure becomes clearer than if the users are integrated as own nodes in a node-link model (see the example in Figure 1). Further, within OrbitFlower and OrbitList the linkage of users to both, roles and organizational units can be perceived at a first glance whereas the node-link model example in Figure 1 conveys an indirect linkage between user and organizational unit via the *user-role* and *role-organizational unit* relations.

Moreover for managing of collaborative tasks, security-relevant information (e.g., is the user linked to the right roles and organizational units) plays an important role and can be visually verified more easily based on the OrbitFlower and OrbitList visualization than based on node-link representations as used in many (commercial) business process modeling and workflow systems. For example, we could observe that participants who were familiar with the organizational structure of the the Faculty of Computer Science at the University of Vienna detected very fast wrong assignments between users to their roles and their organizational units. Furthermore, as addressed in several studies, change of organizational structures might result in security problems [33], [34], [35]. Whereas Weber et al. [33] proposes to prevent unauthorized changes on organizational structures, [34], [35] focuses on controlling the effects of organizational changes with access and authorization rules. Violations of access rules might result from changes that cause empty valid actor sets, i.e., no authorized user can be determined for a certain task/access rule [34]. Violation of authorization constraints might be caused if the number of authorized users drops below a certain threshold, for example, separation of duties with less than two users assigned cannot be fulfilled [35]. In both cases, the OrbitFlower and OrbitList can help to identify possible violation causes after changes immediately, since the visual properties (e.g., color or size) of roles/organizational units indicate that only few or no users are assigned to. The actual number of assigned users can then be quickly looked up by clicking on the respective role/organizational unit. Therefore, the OrbitFlower and OrbitList provide the possibility to visually check effects of organizational changes and hence to prevent security violations.

VI. Related Work

For the visualization of the organizational structure and assignments (e.g., users to roles or permissions to roles) in access control systems, primarily node-link representations are used (see e.g., [4], [10], [12], [14], [29]). For example, Harbach and Smith [29] present an approach – called Mind Mesh – which is inspired by the Mind Maps approach [36] to present organizational information like departments/groups, their projects, users, and resources as nodes and their relationships as links. Furthermore, there exist several works that use the well-known node-link diagrams of the Unified Modeling Language (UML) to visualize the specification of the access control models (see e.g., [5], [37], [38], [39], [40]).

The reason for the popularity of node-link approaches is that they can provide a good overview of the organizational/-role hierarchy and can show the relationships between nodes clearly. Often different colors for links were used to distinguish between different relationship types (see e.g., [4], [10], [29]). However, especially for a large graphs and for large number of connections between the nodes, it can happen that the graph can be cluttered and therefore it is difficult distinguish and to follow the relationships. The approach by Feng et al. [10] uses a double-layered layout to allow users to see the different relationship types not only because of their color-coding but also because of their spatial separation. They use one layer to present the hierarchy of roles and their permissions while the users are visualized on the second layer. The assignments of users to their roles are presented as links between the two layers. However, for large number of assignments of users it can happen that the relationships between the layers are cluttered. To reduce the number of links, our both approaches visualize only the hierarchical relationships and the assignments between roles and organizational units as links. The assignments of users to their roles and organizational units are shown with the help of the brushing and linking technique.

VII. Conclusion

Organizational information is often from interest to communicate and cooperate on common tasks in organizations. Visualizations help to analyze organizational structure and interconnections between entities and groups. The paper presented two visualization approaches, called OrbitFlower and OrbitList, in order to make the dependencies between organizational units, roles, and users recognizable. The aim of both approaches is to support the analysis of information such as the assignments of users to their roles and organizational units. For an effective usage of a visualizations it is necessary that the basic concept is clear and understandable. Therefore, it was important for us to evaluate the design ideas already in the early development phase with experts to get feedback and to identify misinterpretations or unclear elements of both approaches. Despite the small number of participants, we delivered valuable qualitative feedback and inspiration for further development. The results of the evaluation study show us that both visualization approaches got positive reactions. Especially, the available additional information (e.g., number of assigned users, and the distribution of users between roles and organizational units) via different visual properties and the visual separation of users from the organizational structure were stated as very useful.

In further work, we plan to expand the collection of visualization approaches (e.g., spring algorithms, matrix visualization) in order to provide an optimal support for users with the SPRINT Enactment Engine. Moreover, we plan to find solutions to integrate security-relevant information e.g., access rules, permissions information, change information, in our visualization approaches. Finally, we will conduct an extensive evaluation to verify the effectiveness of these approaches with a larger number of users and compare them with other approaches.

Acknowledgements

The research was funded by COMET K1, FFG - Austrian Research Promotion Agency. We thank Kinga Anna Wozniak for modeling the organizational structure as depicted in Figure 1. We also thank Reinhold Dunkl and Günter Wallner for their feedback and our participants of our study for their help and contributions.

References

[1] W. J. Tolone, G.-J. Ahn, T. Pai, and S.-P. Hong, "Access control in collaborative systems," *ACM Comput. Surv.*, vol. 37, no. 1, pp. 29–41, 2005.

[2] P. Inglesant, M. A. Sasse, D. Chadwick, and L. L. Shi, "Expressions of expertness: The virtuous circle of natural language for access control policy specification," in *Proc. of the 4th Symposium on Usable Privacy and Security*. ACM Press, 2008, pp. 77–88.

[3] G. Tassey, M. Gallaher, A. O'Connor, and B. Kropp, "The economic impact of Role-Based access control," National Institute of Standards and Technology (NIST), Tech. Rep. RTI Project Number: 07007.012, 2002.

[4] I. Aedo, P. Diaz, and D. Sanz, "An RBAC model-based approach to specify the access policies of web-based emergency information systems," *The Int'l Journal of Intelligent Control and Systems*, vol. 11, no. 4, pp. 272–283, 2006.

[5] M. E. Shin and G. Ahn, "UML-Based representation of role-based access control," in *Proc. of the IEEE Int'l Workshops on Enabling Technologies*. IEEE Computer Society, 2000.

[6] E. J. Coyne, "Role engineering," in *Proc. of the 1st ACM Workshop on Role-Based Access Control*. ACM Press, 1996.

[7] D. Nadler and M. Tushman, "A model for diagnosing organizational behavior," *Organizational Dynamics*, vol. 9, no. 2, pp. 35–51, 1980.

[8] K. Beznosov, P. Inglesant, J. Lobo, R. Reeder, and M. Zurko, "Usability meets access control: Challenges and research opportunities," in *Proc. of the 14th ACM Symposium on Access Control Models and Technologies*. ACM Press, 2009, pp. 73–74.

[9] A. Colantonio, R. Di Pietro, A. Ocello, and N. V. Verde, "Visual role mining: A picture is worth a thousand roles," *IEEE Transactions on Knowledge and Data Engineering*, 2011.

[10] X. Feng, B. Ge, Y. Sun, Z. Wang, and D. Tang, "Enhancing role management in Role-Based access control," in *Proc. of the 3rd IEEE Int'l Conf. on Broadband Network and Multimedia Technology*. IEEE Computer Society, 2010, pp. 677–683.

[11] C. Giblin, M. Graf, G. Karjoth, A. Wespi, I. Molloy, J. Lobo, and S. Calo, "Towards an integrated approach to role engineering," in *Proc. of the 3rd ACM Workshop on Assurable and Usable Security Configuration*. ACM Press, 2010, pp. 63–70.

[12] L. Hua and S. Osborn, "Modeling UNIX access control with a role graph," in *Proc. of the Int'l Conf. on Computers and Information*, 1998.

[13] P. Rao, G. Ghinita, E. Bertino, and J. Lobo, "Visualization for access control policy analysis results using multi-level grids," in *Proc. of the IEEE Int'l Symposium on Policies for Distributed Systems and Networks*. IEEE Computer Society, 2009, pp. 25–28.

[14] B. Sundaresan, A. Upendra, and R. S. Sundaram, "Detect discrepancy in permission assignments," *Int'l Journal of Computer Applications*, vol. 31, no. 8, pp. 6–13, 2011.

[15] K. Vaniea, Q. Ni, L. Cranor, and E. Bertino, "Access control policy analysis and visualization tools for security professionals," in *Proc. of the 4th Symposium on Usable Privacy and Security*, 2008.

[16] W. Xu, M. Shehab, and G. Ahn, "Visualization based policy analysis: Case study in SELinux," in *Proc. of the 13th ACM Symposium on Access Control Models and Technologies*. ACM Press, 2008, pp. 165–174.

[17] S. Kriglstein and G. Wallner, "Human centered design in practice: A case study with the ontology visualization tool Knoocks," in *Computer Vision, Imaging and Computer Graphics Theory and Applications - VISIGRAPP 2011*. Springer, 2012, in print.

[18] O. A. Kulyk, R. Kosara, J. Urquiza-Fuentes, and I. H. C. Wassink, "Human-centered aspects," in *Human-Centered Visualization Environments*, A. E. A. Kerren and J. Meyer, Eds. Springer, 2006, pp. 13–75.

[19] M. Leitner, J. Mangler, and S. Rinderle-Ma, "SPRINT - Responsibilities: Design and development of security policies in process-aware information systems," *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA)*, vol. 2, no. 4, pp. 1–24, 2011.

[20] B. Shneiderman, "The eyes have it: A task by data type taxonomy for information visualizations," in *Proc. of the 1996 IEEE Symposium on Visual Languages*. IEEE Computer Society, 1996.

[21] J. Lamping, R. Rao, and P. Pirolli, "A focus+context technique based on hyperbolic geometry for visualizing large hierarchies," in *Proc. of the SIGCHI Conf. on Human Factors in Computing Systems*. ACM Press/Addison-Wesley Publishing Co., 1995, pp. 401–408.

[22] B. Shneiderman, "Tree visualization with tree-maps: 2-d space-filling approach," *ACM Trans. Graph.*, vol. 11, pp. 92–99, 1992.

[23] C. Plaisant, J. Grosjean, and B. B. Bederson, "Spacetree: Supporting exploration in large node link tree, design evolution and empirical evaluation," in *Proc. of the IEEE Symposium on Information Visualization (InfoVis'02)*. IEEE Computer Society, 2002.

[24] L. Hong, F. Meng, and J. Cai, "Research on layout algorithms for better data visualization," in *Proc. of the 2nd Symposium Int'l Computer Science and Computational Technology(ISCSCT 09)*. Academy Publisher, 2009, pp. 369–372.

[25] M. Burch, F. Bott, F. Beck, and S. Diehl, "Cartesian vs. radial — a comparative evaluation of two visualization tools," in *Proc. of the 4th Int'l Symposium on Advances in Visual Computing*. Springer, 2008, pp. 151–160.

[26] S. Diehl, F. Beck, and M. Burch, "Uncovering strengths and weaknesses of radial visualizations–an empirical approach," *IEEE Transactions on Visualization and Computer Graphics*, vol. 16, no. 6, 2010.

[27] W. Huang, S.-H. Hong, and P. Eades, "Effects of sociogram drawing conventions and edge crossings in social network visualization," *Journal of Graph Algorithms Applications*, vol. 11, no. 2, pp. 397–429, 2007.

[28] J. Stasko, "An evaluation of space-filling information visualizations for depicting hierarchical structures," *Int'l Journal of Human-Computer Studies*, vol. 53, pp. 663–694, 2000.

[29] M. Harbach and M. Smith, "Visual access control for research ecosystems," in *Proc. of the 5th IEEE Int'l Conf. on Digital Ecosystems and Technologies*. IEEE Computer Society, 2011, pp. 101–108.

[30] D. Stone, C. Jarrett, M. Woodroffe, and S. Minocha, *User Interface Design and Evaluation*. Morgan Kaufmann, 2005.

[31] C. Wilson, *User Experience Re-Mastered: Your Guide to Getting the Right Design*. Elsevier, 2009.

[32] M. Strembeck and J. Mendling, "Generic algorithms for consistency checking of mutual-exclusion and binding constraints in a business process context," in *On the Move to Meaningful Internet Systems: OTM 2010*, ser. LNCS, vol. 6426, 2010, pp. 204–221.

[33] B. Weber, M. Reichert, W. Wild, and S. Rinderle, "Balancing flexibility and security in adaptive process management systems." in *Proc. of the Int'l Conf. on Cooperative Information Systems*, ser. LNCS, Springer, Ed., vol. 3760,, 2005, pp. 59–76.

[34] S. Rinderle and M. Reichert, "A formal framework for adaptive access control models," *Journal on Data Semantics*, no. IX, pp. 82–112, 2007.

[35] S. Rinderle-Ma and M. Leitner, "On evolving organizational models without losing control on authorization constraints in web service orchestrations," in *Proc. of the IEEE 12th Conf. on Commerce and Enterprise Computing (CEC)*. IEEE Computer Society, 2010, pp. 128–135.

[36] T. Buzan and B. Buzan, *The Mind Map Book*. BBC Active, 2003.

[37] A. Cenys, A. Normantas, and L. Radvilavicius, "Designing role-based access control policies with UML," *Journal of Engineering Science and Technology Review*, vol. 2, no. 1, pp. 48–50, 2009.

[38] T. Mustafa, K. Sohr, D. Dang, M. Drouineaud, and S. Kowski, "Implementing advanced RBAC administration functionality with USE," *Electronic Communications of the EASST*, vol. 15, no. 0, 2009.

[39] I. Ray, N. Li, R. France, and D. Kim, "Using UML to visualize role-based access control constraints," in *Proc. of the 9th ACM Symposium on Access Control Models and Technologies*. ACM Press, 2004, pp. 115–124.

[40] K. Sohr, M. Drouineaud, G. Ahn, and M. Gogolla, "Analyzing and managing role-based access control policies," *IEEE Transactions on Knowledge and Data Engineering*, vol. 20, no. 7, pp. 924–939, 2008.