An Intelligent Information Sharing Control System for Dynamic Collaborations

Ahmad Kamran Malik
kamran@infosys.tuwien.ac.at

Schahram Dustdar
dustdar@infosys.tuwien.ac.at

Distributed Systems Group, Institute of Information Systems
Vienna University of Technology
Vienna, Austria

ABSTRACT

Information sharing is a basic requirement in Collaborative Working Environment (CWE), yet privacy of the owner of shared information needs special attention. Personal information and shared information coexist in enterprise-based users, teams, and their activities. Distributed and dynamic collaborations in CWE are a challenge for privacy preservation due to lack of trust among users and thus require an intelligent sharing control policy. Our system intelligently monitors and analysis status of collaborating users, plans sharing control activities according to changing collaborative relationships, user interactions, and context conditions which help in dynamic adaptation of sharing control policy. We present an intelligent sharing control architecture, its implementation, and discussion using Web services and an intelligent context sharing messenger application.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous

General Terms

Human Factors

Keywords

Information privacy, collaborative systems, sharing control policy, intelligent adaptation.

1. INTRODUCTION

In collaborative networks [3], collaborations are mostly distributed and dynamic where collaborative relationships among users (for example, colleague in an enterprise, members in a team, working on common activity) and their context change frequently. This dynamic environment requires intelligent adaptation of collaborative relationships among users and their sharing control policies at runtime. Our system intelligently monitors changing context and collaborative relationships, creates collaboration logs, analysis intelligently whether sharing control policy for personal and shared context needs adaptation to reflect changes in collaborative relationships. In addition, distributed and dynamic users do not have trust relationships, so privacy of their personal information being shared is an issue. We arrange context in different hierarchy levels and owner-defined intelligent sharing policy is used to effectively share context and preserve the user's privacy. Context information related to users, their environment, and their activities is shared with other users to achieve their mutual goals [6]. This context may include information which user perceives as her personal information, for example, her location, resources etc. To allow an owner to control her personal information two types of context are managed in this system; personal context, for example, user's location and her personal resources and activities, and shared context, for example, current activities in a team, activity status and details. Personal context is managed by user (who is owner of context) and shared context is managed by the team leader who collects it from users that are performing activities. Context in our system is arranged in different hierarchy levels to share different level of context with user having different collaborative relationships.

Our system uses two types of sharing control policies, enterprise-defined policy and owner-defined policy. Enterprise-based sharing control policy is based on roles assigned to users by enterprise whereas owner-defined policy is defined and can be adapted either by owner of context manually or intelligently by her system. Intelligent sharing control mechanism is able to detect changes in collaborative relationships of owner with other users and performs dynamic adaptation in owner-defined sharing control policy, for example, access control feedback provided in [9]. Manual changes are needed only when the owner of context wants to change her sharing control policy with one or more existing or new users. We describe the intelligent sharing control architecture of our system and explain working of its components. Our intelligent sharing control system is implemented in Java and uses a messenger application in which users can pose queries for sharing context using Web service calls. Intelligent adaptation mechanism periodically monitors, analysis, and adapts collaborative relationships and owner-defined policy, and an owner can share her context with others users in controlled manner, only allowing others to access her context at a certain level of granularity. Our contributions include: intelligent adaptation of sharing control policy at runtime, privacy

FIT'10, December 21-23, 2010, Islamabad, Pakistan.

preservation of owner's personal context and owner-defined sharing control policy.

The remainder of the paper is organized as follows. Section II describes collaboration-based intelligent sharing control. Section III gives the intelligent sharing control and dynamic adaptation. Section IV shows the intelligent sharing control architecture. Section V contains implementation and discussion. Section VI gives the background and related work. Section VII contains acknowledgements and Section VIII concludes the paper and highlights future work.

2. COLLABORATION-BASED INTELLIGENT SHARING CONTROL

In this section, we describe a scenario related to collaboration-based intelligent sharing control. Our scenario is based on Collaborative Working Environment (CWE), for example, [1]. This is a distributed and dynamic environment where enterprises collaborate and create virtual teams of experts and select members based on their skills and expertise. A user being member of one or more teams performs activities assigned by team leader from a local or remote site. Users and their resources can be distributed over long distances connected via the Internet using Web services. Users as well as teams are dynamic which means that a user can join or leave a team at any time. Teams are also overlapping, a user can be part of more than one teams at a time. Users need to share context information related to other users as well as activities and teams to fulfill their ongoing activities in a dynamic collaborative environment. For this reason, context is stored in user's personal site as well as team leader's site. Context is divided into personal and shared context. Personal context is stored at each user's device whereas shared context is stored at user's device as well as team leader's device. This is because shared context consists of two types: first which is related to activities currently being performed by users, and second which is related to past activities is stored at team leader's device. Two types of services are used to access these contexts which are personal services and shared services. Sharing control policy at user's site and team leader's site is used to control the sharing of personal and shared context. Two types of sharing control policies are used: first is called user-defined policy is used to control the sharing of personal context, and second is called enterprise-defined policy which is used to control the sharing of shared context related to team and their activities.

Figure 1(a) shows an example of our scenario. Enterprises *E1* and *E2* create two teams *T1* and *T2*. Teams *T1* and *T2* are controlled by a team leaders *TL1* and *TL2* respectively. User *U1* is member of team *T1*, *U2* is member of team *T2* whereas user *U3* is member of both teams *T1* and *T2* at a time (resulting in overlapping teams). Each user's system consists of a user, activities assigned to user, context store of the user, sharing control policy for user's personal context and shared context, and Web services which are used by other users to make a context sharing request. Activities of the team are assigned to each team member by their team leaders. User *U1* and *U3* are involved in a mutual activity *A1* of the team *T1* while user *U2* and *U3* are members of team *T2* but do not share an activity of their team. Each user's context store contains her personal context (for example, location, devices) as well as shared context (for example, assigned activities, activity status). Context is managed at

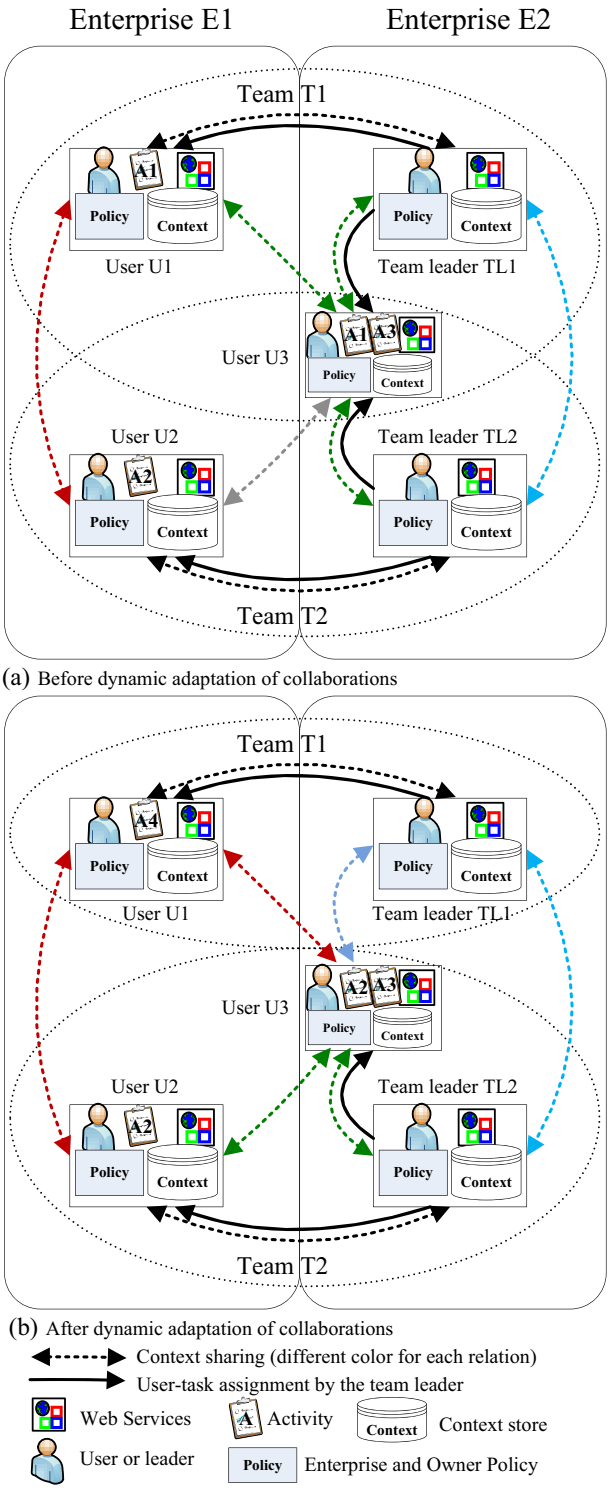

Figure 1: Dynamic collaborations and intelligent sharing control

different levels in context stores to share it with different collaborating partners (for example, three levels for location context are room number, street address, and city). Every user can send a request to other user for context sharing. Policy at each user system describes rules and constraints for sharing context with other users. Policy describes which context at what level can be shared with whom in certain context conditions.

Context sharing in our system is based on collaborative relationships among users and is depicted by different colored arrows for each type of relationship in Figure 1(a). For example, context at highest level of granularity can be shared with users who are working in a mutual activity (green arrow showing highest level of context sharing among users *U1* and *U3* involved in mutual activity *A1*), at a smaller level with the users in different activities but same team (users *U2* and *U3* are involved in different activities but same team *T2* shown by grey arrow), and even less sharing with users working in different activities as well as different teams (user *U1* and *U2* are involved in different activities as well as different teams are shown by red arrow, *TL1* and *TL2* are team leaders of different teams shown by blue arrow, and relationship in *TL1* and *U1* shown in black arrow as they are in same team but different enterprises). Additionally, as it is a dynamic environment, changes can happen in user's context as well as their activity, role, or team assignments. An intelligent system automatically monitors and analysis the changes and update sharing control policy. Figure 1(b) shows the changed scenario when activity *A1* is finished and a new activity *A4* is assigned to user *U1* by team leader *TL1*. Also user *U3* is asked by team leader *TL2* to help user *U2* in activity *A2*. Now user *U3* has both activities related to team *T2*, so she leaves team *T1*. Team *T1* and *T2* are no more overlapping and all relationships of user *U3* are now changed except one with team leader *TL2*. Due to the changes in activity assignments collaborations are changed which result in change of sharing levels among users (changed sharing level shown by green arrow among user *U2* and *U3*, red arrow among users *U1* and *U3*, and light blue among *TL1* and *U3* as shown in Figure 1(b)).

3. INTELLIGENT SHARING CONTROL AND DYNAMIC ADAPTATION

Sharing control is used to control sharing of information among collaborating users. In contrast to the term *access control* which tries to control access of a subject to an object, *sharing control* emphasizes the requirement to include relationships among users and history of their collaborations for controlling information sharing.

Collaborative relationships exist among users as described in our scenario above, for example, users are colleague in an enterprise or they are members of same team or involved in a mutual activity. Having any one or more of these relationships users can share context at certain level. What level of context should be allowed to a certain user depends on her relationship and collaboration history with other user. In addition, user's role in enterprise and current context conditions are used as constraints to determine the level of context sharing. For dynamic adaptation of collaborative relationships and policies our system intelligently uses provided knowledge and autonomic cycle of monitoring, analysis, plan, and execution of runtime adaptation as shown in

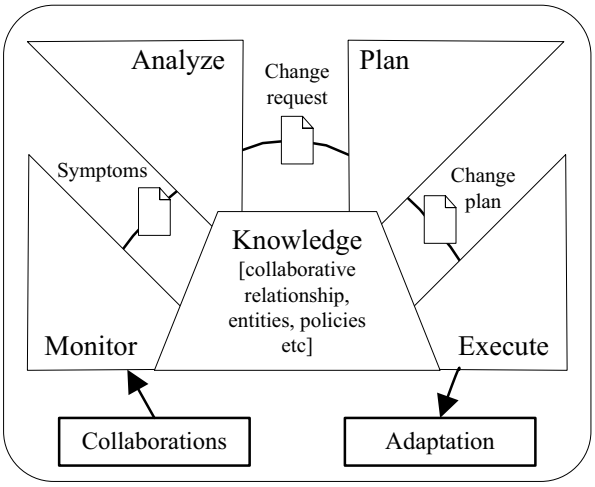

Figure 2: Dynamic collaboration and adaptation, *adapted from [2]*

Figure 2.

On one hand our system determines level of context sharing using collaborative relationships, on the other hand system intelligently monitors and dynamically adapt owner-defined sharing control policy based on updated relationships. Our system uses two types of sharing control policies: enterprise-defined role-based policy and owner-defined rule based policy. Owner is not allowed to change the enterprise-defined policy which is rather static and is based on roles of users in enterprise. Owner-defined policy can be changed by the owner whenever she wants to give special sharing permission to a collaborating user, activity members or team, to a specific level of context detail. An example of owner-defined policy is described in Listing 1.

This example shows the sharing control policy defined by an owner of context, for example, user *U2* shown in Figure 1. The policy describes that a user of team *T2* can access *activity A2 service* of user *U2* as long as activity *A2* is not finished. In addition it says that users who are collaborating in same activity *A2* (Mu for Mutual) will get service at level *L1* while others members of team *T2* (NMu for Non-Mutual) will get service details at level *L2*. Level *L1* describes the lowest level of detail while level *L2* describes a higher level of granularity. In the same manner policy can be described for all entities and at any level of detail with positive (+) as well as negative (-) sharing rules. Our system intelligently monitors collaboration changes and logs any changes in the files.

Interactions among users performing their activities within and across teams and enterprises are captured. Similarly collaborative relationships among users and their activity progress are monitored. In this way, data collected from all users is filtered and aggregated to generate interaction details at different entity levels of our complex scenario. For example, for privacy preservation of a user's personal context, her owner-defined sharing control rules are adapted based on change in the user's relationships with other users, their current context, and current interactions. At a higher level, aggregated monitored data helps in determining level of information sharing by a single user, or level of collected

```
<OwnerRules>
  <Rule>
    <Action="+">
      <Subject>
        <Predicate>
          <Op>eq</Op>
          <EntityName> team </EntityName>
          <EntityFunc> name </EntityFunc>
          <EntityValue>T2</EntityValue>
        </Predicate>
      </Subject>
      <Object> activity A2 service </Object>
      <Condition>
        <Predicate>
          <Op>eq</Op>
          <EntityName> activity </EntityName>
          <EntityFunc> name </EntityFunc>
          <EntityValue>A2</EntityValue>
        </Predicate>
        <Exp>AND</Exp>
        <Predicate>
          <Op>neq</Op>
          <EntityName> activity </EntityName>
          <EntityFunc> status </EntityFunc>
          <EntityValue> finished </EntityValue>
        </Predicate>
      </Condition>
      <AccessLevel>
          <Predicate>
            <Op>eq<Op>
            <EntityName> Collaboration </EntityName>
            <EntityValue> Mu </EntityValue>
          </Predicate>
          <Level> L1 </Level>
          <Predicate>
            <Op>eq<Op>
            <EntityName> Collaboration </EntityName>
            <EntityValue> NMu </EntityValue>
          </Predicate>
          <Level> L2 </Level>
      </AccessLevel>
    </Action>
  </Rule>
</OwnerRules>
```

Listing 1: Example of Owner-defined sharing control policy

information sharing within an activity, team, or enterprise. By analyzing this information hidden sharing relations become visible and it can be seen which users, teams, or enterprises are not effectively sharing required level of context with their collaborating partners. Planning step uses policies specified by the enterprise to achieve defined goals. Analyzed data helps in planning phase to take necessary action for sharing required level of context information to achieve desired goals. For example, necessary changes can be suggested to administrator of enterprise-defined policy for sharing context at certain level. These results can also be used by the team leader to find users who are sharing their activity and team related context with others at a lower or higher level of detail than required which can result in inefficiency or violation of privacy requirements defined by enterprise.

4. INTELLIGENT SHARING CONTROL ARCHITECTURE

Our Sharing Control architecture is shown in the Figure 3. The user system represents a user, her devices, services, context and policies. Team leader represents the team and contains information about team members, activities, resources, and context related to team activities and resources. The enterprise system represents an enterprise which creates teams and provides users, resources, and policies to them. We describe components of our sharing control architecture and their interconnections in the following paragraphs.

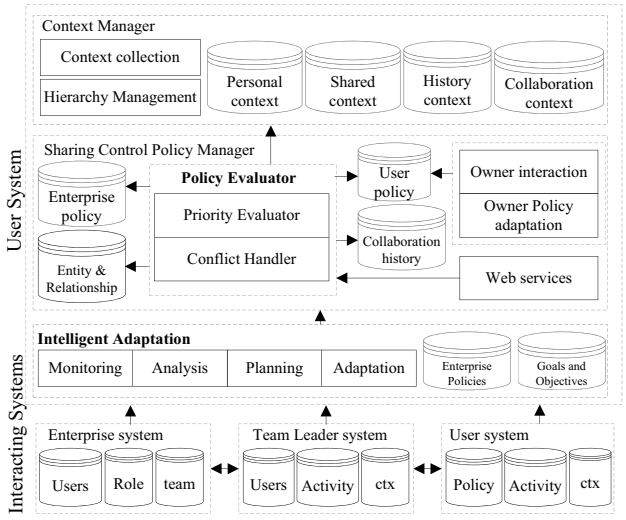

Figure 3: Intelligent Sharing Control Architecture

4.1 Intelligent adaptation

Intelligent adaptation is core part of the architecture which intelligently monitors, analyze, plan, and adapt sharing policy based on current context conditions, collaborative relationships, and history of previous sharing. Whenever a change is detected in status of any one of the entities in our system, this component takes action and updates the sharing control policy, for example, owner-defined sharing control policy updated in our scenario from Figure 1(a) to Figure 1(b) represented by different colors of arrows. This component can detect sharing agreement violations (for example, collaborating user is not providing agreed context), and also upon detecting a major change in collaborative relationship with a user, it can cancel or at least deactivate sharing control rule with the user and send an information message to the owner of system.

4.2 Priority and conflict handling

Priority and conflict handling component resolves any conflicts occurring in sharing control policy. If two rules are conflicting for a sharing control decision, for example, one rule says users of team *T1* cannot get access to personal context services, and other rule says that users involved in activity *A1* can get access to personal context. In this case, priority handling component decides in favor of smaller entity rules, for example, activity scope is smaller than team, so sharing is allowed. In addition, if entities are of same scope then negative sharing rule gets priority over positive sharing rule.

4.3 Sharing control policy

Our sharing control policy is based on enterprise-defined role-based sharing control policy and owner-defined sharing control policy. Enterprise defines policy based on roles assigned to its employees. This policy is rather static because roles in enterprise are of static nature. Roles of employees in

an enterprise do not change for long time. As our scenario is dynamic sharing control where status of any entity in the system including users can change at any time, so we require a dynamic policy in presence of enterprise-defined rules. To allow a user (who is owner of her personal context) to control her sharing control policy we define owner-defined policy rules. Owner-defined policy overrides enterprise-defined policy as far as the personal context sharing is concerned. An owner can interact with her own policy and can adapt it whenever required. For shared context which is related to team and its activities, enterprise-defined sharing control policy is used.

4.4 Context Manager

This component provides context information to the policy management component. Context data can be collected from sensors and also from the manual input. Context is provided by context servers from different sensors in the environment. Activity related context is also provided by manual entry by individuals involved in the activity. Context is arranged into different hierarchies so that it can be shared with users having different collaborative relationships and is saved into context store. Context serves multiple purposes in our system. Context is used to share context information with collaborating partners and is also used to evaluate context sharing requests using context constraints. Following types of context features are used in our intelligent sharing control system.

- Personal context include user and her environment related features. For example, user's location, resources etc.
- Shared context consists of activity and team related features, for example, current activities, activity status, scheduled activities, calendar of activities.
- Collaborative relationships, for example, collaborating users in an activity, team, or enterprise, collaboration level.
- History of collaboration related features, for example, sharing history, number of accesses, type of sharing.

Users requesting for context cannot be allowed to access all context information related to that particular service. A context owner may require that users having specific role should be allowed to access more information than others. This system uses hierarchies for context information by storing all context information at three levels *L1*, *L2* and *L3*. Level *L1* describes the most detailed information and *L3* describes least detailed information.

4.5 Context Sharing: Query and Subscribe

An individual can use Query and subscribe to access context information from other collaborating users. Query can be directly sent to some known online user by calling its related Web service. An individual can subscribe for certain type of context during a mutual activity with the other member of her team. If the requester is allowed to access requested context service, requested system asks context servers for requester's current context required by the context constraints in owner-defined policy. After validation of constraints, the level of access for the requester is determined as described in the sharing control policy depending on requester's collaborative relationship with the requested user.

5. IMPLEMENTATION AND DISCUSSION

Our sharing control system has been implemented using Web services in Java. Enterprise-defined policy and owner-defined policy are described in XML. Context store is a database which contains different types of personal and shared context in the form of hierarchies of three levels. Web services are used for communication among users. A user can call a required Web service provided by other users to send a context sharing request. The requested system can automatically check for authorizations and respond to the requested system.

We created messenger application to test our sharing control policy. It contains list of buddies who can share each other's context by using services provided by online members of their community. Whenever a sharing request is made, the requester and the requested system make an entry in their interaction log files which contains history of their collaboration, relationship types etc. After certain interval of time user's system automatically monitors changes in collaboration and relationships and performs dynamic adaptation of owner-defined context sharing policy for the users whose collaborative relationship status with the owner of monitoring system is changed. User is also allowed to add or delete new sharing control rules manually if she wants to share context with certain user or group of users at a certain level of granularity. In a similar manner, a team leader can monitor sharing among team members belonging to similar or different enterprises and can adapt sharing control policy at a higher level, for example, enterprise-defined sharing control policy instead of user-defined sharing control policy. In this way our sharing control system exhibits its control over the sharing of personal as well as shared information in a collaborative environment by intelligently using knowledge of entities in the system, their collaborative relationships, context sharing interaction, and sharing control policies.

6. BACKGROUND AND RELATED WORK

Context represents information about users, their environment and activities which is useful when it is managed in a form such that collaborating users can get knowledge about shared activities. This context information can be shared among collaborating users in networked enterprises [3][13]. Sharing of information is a challenging task which involved many problems and risks about the data being shared [17]. Our focus is privacy of personal context and privacy of team related shared context in dynamic collaborative environments [7]. Our idea of *sharing control* includes control of information being shared as well as sharing of control among enterprise and user. Whereas *access control*, for example Role-based Access Control (RBAC) [15], only includes control of information being accessed. The requirements of access control in collaborative environments are described in [16] and a survey is presented in [18]. Due to the dynamic nature of such collaborative systems, the status of entities, and their context in such an environment changes frequently which results in their collaborative relationships with other entities. Our system tries to handle such changes using intelligent monitoring and analysis of the collaborative environment, for example, the feedback for access control decisions

presented in [9]. Additionally, like autonomic systems [11], our system plans required changes in the collaborative relationships and related sharing policies which are dynamically adapted at runtime. Some other systems like [10] and [19] perform dynamic adaptation of policies but are not based on autonomic principles.

We use hybrid sharing control policy which consists of enterprise-defined sharing control policy and owner-defined sharing control policy. Enterprise-defined policy is based on RBAC [15] while owner-defined policy is rule-based which includes context conditions as constraints for sharing control rules. RBAC model can assign many permissions to many role, many roles to many users, so it is a very efficient model for access rights management at large scale. Yet it does not fulfil the requirements of dynamic collaborative systems like [1] because it is a type-based access control using roles which lacks access control at an individual level. We make use of context in our system for sharing context as well as for restricting the sharing of context using context constraints while RBAC model [15] does not make use of context for these purposes. Some other systems which made use of context in RBAC include [4] and [14]. Hybrid policy used in our system may result in policy conflicts which are handled by defining priorities for different entities and different types of rules. Strategies for rule conflict handling are described in [8] and [5]. Our system [12] describes a conflict handling mechanism for dynamic collaborative environments using collaborative relationships and priorities of all entities in the environment.

7. ACKNOWLEDGEMENTS

This work is partially supported by the European Union through the FP7-216256 project COIN.

8. CONCLUSION AND FUTURE WORK

In this paper intelligent sharing control is described for distributed collaborating users. The architecture of our system is described which uses intelligent monitoring, analysis and adaptation to share context among collaborating users at a certain level of granularity and preserves user's privacy requirements. Collaborative relationships and collaborative interaction is monitored to dynamically analyze relationships and interactions and adapt owner-defined sharing control policy if required. Context of different types is used to share information of user and their environment among collaborating partners and it is also used to control sharing of context using context constraints. Our future work includes the use of semantic techniques for context management, service discovery, and context sharing.

9. REFERENCES

[1] European union project coin. http://www.coin-ip.eu.

[2] An architectural blueprint for autonomic computing, IBM White Paper, 2005.

[3] L. M. Camarinha-Matos and H. Afsarmanesh. Collaborative networks: a new scientific discipline. *Journal of Intelligent Manufacturing*, 16(4):439–452, 2005.

[4] M. J. Covington, W. Long, S. Srinivasan, A. K. Dey, M. Ahamad, and G. D. Abowd. Securing context-aware applications using environment roles. In *SACMAT '01*, pages 10–20. ACM, 2001.

[5] F. Cuppens, N. Cuppens-Boulahia, and M. B. Ghorbel. High level conflict management strategies in advanced access control models. *Electron. Notes Theor. Comput. Sci.*, 186:3–26, 2007.

[6] A. K. Dey and G. D. Abowd. Towards a better understanding of context and context-awareness. In *Workshop on The What, Who, Where, When, and How of Context-Awareness, (CHI 2000)*, APRIL 2000.

[7] S. Dustdar. Caramba – A process-aware collaboration system supporting ad hoc and collaborative processes in virtual teams. *Journal of Distributed and Parallel Databases*, 15(1):45–66, JANUARY 2004.

[8] S. Jajodia, P. Samarati, V. S. Subrahmanian, and E. Bertino. A unified framework for enforcing multiple access control policies. *SIGMOD Rec.*, 26(2):474–485, 1997.

[9] A. Kapadia, G. Sampemane, and R. H. Campbell. Know why your access was denied: regulating feedback for usable security. In *CCS '04: 11th ACM conference on Computer and communications security*, pages 52–61, New York, USA, 2004.

[10] Y.-G. Kim, C.-J. Moon, D. Jeong, J.-O. Lee, C.-Y. Song, and D.-K. Baik. Context-aware access control mechanism for ubiquitous applications. In *Advances in Web Intelligence, AWIC 2005*, volume 3528, pages 236–242, 2005.

[11] H. Koshutanski and F. Massacci. Interactive access control for autonomic systems: From theory to implementation. *ACM Trans. Auton. Adapt. Syst.*, 3(3):1–31, 2008.

[12] A. K. Malik and S. Dustdar. A hybrid sharing control model for context sharing and privacy in collaborative systems (accepted). In *WAMIS 2011, Singapore*, March, 22-25, 2011.

[13] A. K. Malik, H. L. Truong, and S. Dustdar. DySCon: Dynamic sharing control for distributed team collaboration in networked enterprises. In *IEEE Conference on Commerce and Enterprise Computing, CEC 2009, Vienna, Austria*, pages 279–284, 2009.

[14] S.-H. Park, Y.-J. Han, and T.-M. Chung. Context-role based access control for context-aware application. In *High Performance Computing and Communications*, volume 4208 of *LNCS*, pages 572–580, 2006.

[15] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman. Role-based access control models. *IEEE Computer*, 29(2):38–47, Feb. 1996.

[16] H. Shen and P. Dewan. Access control for collaborative environments. In *Proceedings of ACM CSCW'92 Conference on Computer-Supported Cooperative Work*, pages 51–58, 1992.

[17] K. Smith, L. Seligman, and V. Swarup. Everybody share: The challenge of data-sharing systems. *IEEE Computer*, 41(9):54–61, 2008.

[18] W. Tolone, G.-J. Ahn, T. Pai, and S.-P. Hong. Access control in collaborative systems. *ACM Computing Surveys*, 37(1):29–41, Mar. 2005.

[19] A. Toninelli, R. Montanari, L. Kagal, and O. Lassila. A semantic context-aware access control framework for secure collaborations in pervasive computing environments. In *The Semantic Web - ISWC 2006*, volume 4273 of *LNCS*, pages 473–486, 2006.