Context-Aware Management Domains

Ricardo Neisse[1], Maarten Wegdam, Patrícia Dockhorn Costa, and Marten van Sinderen

CTIT, University of Twente, The Netherlands
{r.neisse, m.wegdam, p.dockhorncosta, m.j.vansinderen}@utwente.nl

Abstract. In this paper we extend the concept of management domains to a new concept called Context-Aware Management Domains (CAMDs). CAMDs enable context-aware management of policies allowing the grouping of entities based on context information. Since context is dynamic, so is the domain membership. As a consequence, the association of policies with the entities in the domain also becomes dynamic. In this paper we provide CAMD examples and an information model together with a discussing on our ongoing implementation for our target context-aware service platform.

1 Introduction

Context aware services adapt themselves to the current user's situation. An example of this is a tourist service which uses the current user location, activity, and preferences to personalize tourist advices. In order to support context awareness, service platforms have been designed to support context information acquisition, reasoning and distribution [1].

Typical context-aware service platforms have thousands or millions of entities (users, service providers, context providers, etc.) and different types of policies have to be managed. Policies are required, for instance, to control access to context information, to enforce user's privacy, and to manage trust relationships among the entities. Due to the complexity, dynamicity, and large number of entities, the specification of these policies can easily become unmanageable.

Standard policy management tools ease the policy management, however, the problem with these tools is that they provide either static management capabilities (e.g. management domains [2]), or, if there is some form of dynamic management, this is limited to one specific area (e.g., X-RBAC [3], [4], and [5] for access control and COMITY [6] for trust management). For this reason, these policy management tools do not fulfill the dynamic policy requirements of context-aware service platforms.

In a context-aware service platform, policies are defined based on the context of the entities. One example is a privacy policy stating that "Bob's identity should not be anonymized for nearby persons". In this case, "nearby persons" refers to a set of

[1] Supported by CNPq Scholarship – Brazil

entities not known at policy specification time, because it is not possible to determine beforehand which entities are likely to approach Bob.

In this paper we address context-aware policy management by extending management domains to a new concept called Context-Aware Management Domains (CAMDs). CAMDs are management abstractions that provide dynamic grouping of entities based on common context situations and, as a result, context-aware management of different types of policies. We provide an information model for CAMDs and discuss an implementation strategy for CAMDs in the scope of our target context-aware service platform [1].

This paper is organized as follows. Section 2 introduces our context-aware service platform and describes the policy deployment scenario with examples. Section 3 presents our new concept called Context-Aware Management Domains, the information model, and our ongoing implementation efforts. Section 4 compares our work with related work on context-aware management tools and Section 5 ends this paper with conclusions and future work.

2 Policy Management in a Context-Aware Service Platform

Figure 1 presents our target context-aware service platform considering a single administrative domain and illustrates the main roles we distinguish regarding the platform *management* and *operation* layers.

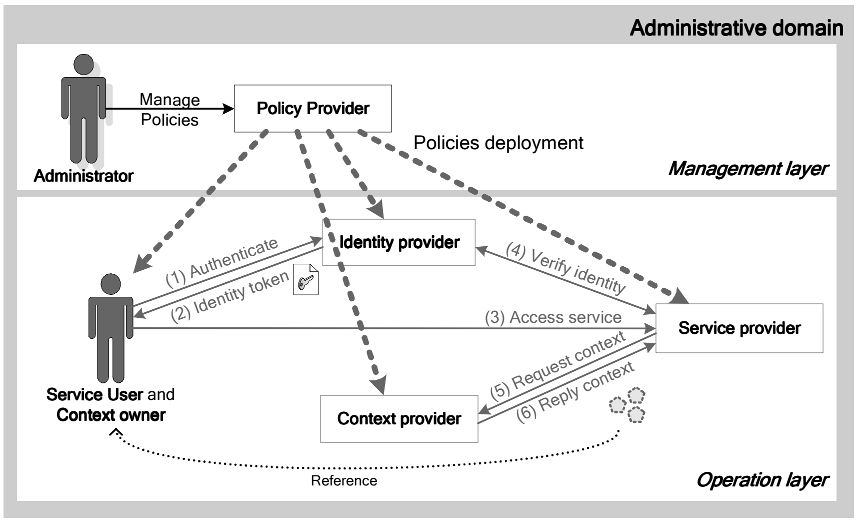

Fig. 1. Policy deployment in a context-aware service platform

Within the *operation layer* a user receives an authentication token (2), after authenticating with an identity provider (1), which is used to access a service provider (3). The service provider verifies the user's identity (4) and retrieves context information to adapt the service (5 and 6). This information can be, for instance, the current activity or location of the user; however, it can also include context information about other entities (context owners) that are relevant for the context-aware service used (e.g. service provider). For details about the operation layer see [1].

Within the management layer an administrator accesses the policy provider in order to manage operation policies. Policies are rules that define a choice in the behavior of the system and can be of different types such as obligation, authorization, refraining, filtering, delegation, and meta-policies [7]. Policies of different types can also focus on different management areas, for example, access control, privacy enforcement and trust management. In this paper we support context-aware management of policies of different types and different areas, however, due to space limitations, we only exemplify obligation policies focusing on privacy enforcement.

Obligation privacy policies describe actions that subjects must perform on target entities under certain conditions [7]. Such a policy could state, for instance, that "15 minutes after getting Bob's location, Alice (Bob's colleague) should delete it" or "Bob's identity should be anonymized by the identity provider when provided to Alice". In the examples above the policy subjects and targets (Bob, Alice, and Bob's identity provider) are individually specified in the policies and do not easily allow Bob to specify policies for a set of entities, for instance, all his colleagues.

In order to allow the deployment of policies for a collection of entities, as opposed to individual entities, management domains [2] can be used as a grouping abstraction (e.g. Bob's colleagues). Management domains reduce the management complexity in large systems because it is hard to specify and apply policies individually for each entity on a large scale. However, one problem with management domains is that they are static, and the inclusion and removal of entities from a domain must be done manually. In this paper we go one step further by defining management domains based on context situations [8] in a new concept called Context-Aware Management Domains (CAMDs).

3 Context-Aware Management Domains

In order to illustrate our new concept we present in Figure 2 an example where the Context-Aware Management Domain (CAMD) "colleagues currently at work" is mapped to every colleague in which the current activity status is "at work". When persons change their activities domain membership also changes. As a result, any association of domain policies with entities also changes, as entities can leave/enter the domain dynamically according to changes in the context information.

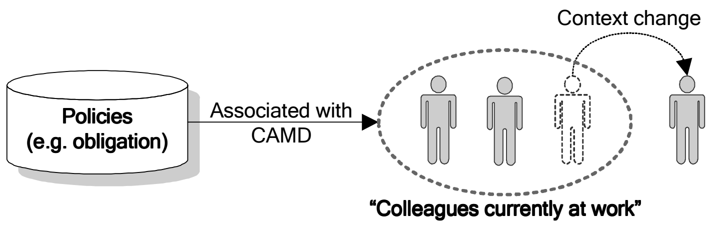

Fig. 2. Change in domain membership from context changes

Through the definition of CAMDs it is possible to associate policies to dynamic sets of entities based on context situations. This has the potential to provide a more flexible management tool for association of policies with entities in a context-aware service platform. Figure 4 presents our information model for CAMDs which combines the policy management model from [2] and the context information model from [8].

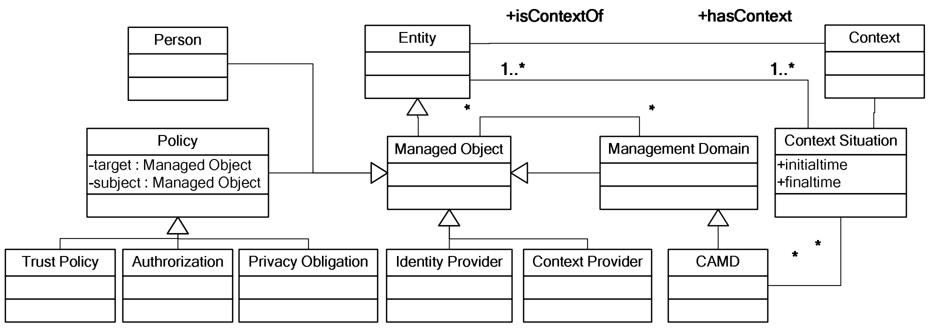

Fig. 3. Information Model for Context-Aware Management Domains (CAMDs)

In [2], a Policy is a managed object triggered by events, which are related to other managed objects namely policy subjects and targets. Policy subjects and targets can be specified individually as individual managed objects or by means of management domains, which are static sets as has already been exemplified. In our model we define a CAMD as a specialized type of a management domain which is related to context situations and derives the entity set from the entities in the associated context situation.

Context situations [7] are composite classes of entities and their context information. A situation has duration, which is defined by the moment the situation begins to hold (initial time), and the time the situation seizes to hold (final time). In this way, the membership of entities is dynamic as they join and leave the domain according to changes in the context situation. We are currently using the approach presented in [7] to define and realize context situations.

An example of a context situation would be "persons nearby", where "nearby" means persons within 20 meters from each other. When two persons are within 10 meters from each other, they are said to be in the “persons nearby” situation. An example of CAMD characterized by the “persons nearby” context situation would be composed of all persons that are currently in situation “persons nearby” and a privacy obligation policy could be specified stating that “when Bob’s identity is requested by nearby persons it should not be anonymized”.

Regarding the implementation of CAMDs we have analyzed the available context management mechanisms supported by our target context-aware service platform [1]. Our platform supports query-response, subscribe-notify, and event-condition-action (ECA) rules. These mechanisms have a direct impact on the performance of the CAMD implementation because it is necessary to query context from distributed context providers in the network to detect changes in the context situations.

We choose to use ECA rules and we are currently implementing CAMDs using a distributed ECA rule engine called D-JESS [9]). D-JESS provides efficient rule execution considering detection of context situations based on a network of context providers. In this way we believe to achieve scalability and low response time in the deployment of policies using CAMDs.

4 Related Work

The work done by Damianou et al. [2] in the Ponder toolkit provides specification of policies and management domains. Entities are statically associated with domains which are then associated with different types of policies. Our work is inspired by their vision of policy-based management using domains however we include context situations as a dynamic component for domain membership management.

Bathi et al. [3] have extended the Role Based Access Control (RBAC) standard to support the definition of parameterized access control roles. Their proposal is called X-RBAC and provides dynamic management of access control roles based on time and location constraints. Their focus is specific in access control policies for XML document sources at different levels (conceptual, schema, XML instance, and element). Compared to our work their work is more restricted to the type of policies (only access control) and type of context (only location and time).

Corradi et al. [5] use context information to adapt trust relationships for pervasive environments in the so called COMITY security model. In their work they associate trust degrees with context conditions which are further associated with authorization and refrain policies allowing dynamic management. As was the case with [3], our concept of a context-aware domain can be seen as a generalization of [5] because our work allows context-aware management of different types of policies and does not focus on a specific management area such as access control or trust management.

5 Conclusions

This paper present a new concept called Context-Aware Management domains (CAMDs) in order to have a flexible and dynamic policy management model for context aware service platforms. In order to realize this new concept we present an information model, based on previous work on context modeling and policy management, and our on going CAMD implementation efforts using a distributed rule engine called D-JESS. Our CAMD information model allows context-aware management of policies in a generic and flexible way and does not limit policies to an specific type or area (e.g. access control and trust).

As future work we plan to finish our implementation using a distributed ECA rule engine and evaluate the scalability and performance regarding response time and network bandwidth consumption. We also want to analyze policy conflicts for CAMDs and study the deploying of CAMDs in a multi-administrative domains environment when the trustworthiness of the context information used for context situations detections is an issue.

Acknowledgements

This work is part of the Freeband AWARENESS project (http://awareness.freeband.nl). Freeband is sponsored by the Dutch government under contract BSIK 03025.

References

1. van Sinderen, M.J.; van Halteren, A.T.; Wegdam, M.; Meeuwissen, H.B.; Eertink, E.H. Supporting Context-aware Mobile Applications: an Infrastructure Approach. IEEE Communication Magazine, Sep. 2006 issue.
2. Damianou, N.;Dulay, N.; Lupu, E.; Sloman, M.; Tonouchi, T. Tools for Domain-based Policy Management of Distributed Systems. IEEE/IFIP NOMS 2002.
3. Joshi, J. B. D.;Bhatti, R.; Bertino, E.; Ghafoor, A. Access-Control Language for Multidomain Environments. IEEE Internet Computing Nov./Dec. 2004.
4. A. Corradi, R. Montanari, D. Tibaldi Context-based Access Control for Ubiquitous Service Provisioning. COMPSAC 2004, Hong Kong, Sep. 2004.
5. Michael J. Covington , Wende Long , Srividhya Srinivasan , Anind K. Dey , Mustaque Ahamad , Gregory D. Abowd, Securing context-aware applications using environment roles. ACM symposium on Access control models and technologies, May 2001, United States.
6. Corradi, A.; Montanari, R.; Tibaldi, D. Context-driven Adaptation of Trust Relationships in Pervasive Collaborative Environments. SAINT 2005, IEEE CS Press.
7. Damianou, N.; Dulay, N.; Lupu, E.; Sloman, M. The Ponder Specification Language. Workshop on Policies for Distributed Systems and Networks (Policy2001), Jan 2001.
8. Dockhorn Costa, P., Guizzardi, G., Andrade Almeida, J.P., Ferreira Pires, L., van Sinderen, M., Situations in Conceptual Modeling of Context. In VORTE 2006 at IEEE EDOC 2006, IEEE Computer Society Press.
9. Cabitza, F., Sarini, M., Dal Seno, B.: DJess - a context-sharing middleware to deploy distributed inference systems in pervasive computing domains. In: Proceeding of International Conference on Pervasive Services (ICPS '05), IEEE CS Press (2005) 229–238