# Modelling context-aware RBAC models for mobile business processes

## Sigrid Schefer-Wenzl*

Institute for Information Systems and New Media,
WU Vienna, Austria
and
Competence Center for IT-Security,
University of Applied Sciences Campus Vienna,
Vienna, Austria
Email: Sigrid.Schefer-Wenzl@fh-campuswien.ac.at
*Corresponding author

## Mark Strembeck

Institute for Information Systems and New Media,
WU Vienna, Austria
Email: Mark.Strembeck@wu.ac.at

**Abstract:** In a mobile computing environment, distributed business processes are executed in varying contexts. Context-aware access control mechanisms help to protect sensitive data and services in mobile application scenarios. Context constraints are a means to consider context information in access control decisions. In this paper, we integrate context constraints with process-related role-based access control (RBAC) to support the secure and context-dependent task execution. In particular, we specify a formal metamodel for process-related and context-aware RBAC models. Subsequently, we define a domain-specific extension for UML Activity diagrams that enables the integrated modelling of context constraints and business processes. In addition, we implemented a software platform that enables the specification and enforcement of process-related context-aware RBAC policies.

**Keywords:** access control; business process modelling; context constraints; object constraint language; security; unified modelling language.

**Reference** to this paper should be made as follows: Schefer-Wenzl, S. and Strembeck, M. (2013) 'Modelling context-aware RBAC models for mobile business processes', *Int. J. Wireless and Mobile Computing*, Vol. 6, No. 5, pp.448–462.

**Biographical notes:** Sigrid Schefer-Wenzl is a researcher and Lecturer at the Competence Center for IT-Security at the University of Applied Sciences Campus Vienna and the Vienna University of Economics and Business (WU Vienna). She received a PhD, a Master, and a Bachelor degree in Information Systems as well as a Master degree in Business Administration from the WU Vienna. Most of her current research focuses on the fields of security and business process modelling.

Mark Strembeck is an Associate Professor of Information Systems at WU Vienna, Austria. He received his Doctoral degree as well as his Habilitation degree (venia docendi) from WU Vienna. He is a key researcher at the Secure Business Austria (SBA) Research and Competence Center, and the Vice Institute Head of the Institute for Information Systems at WU Vienna. His research interests include access control, role engineering, secure business systems, model-driven software development, and business process management.

*This paper is a revised and expanded version of a paper entitled 'Modeling context-aware RBAC models for business processes in ubiquitous computing environments' presented at 'Proceedings of the 3rd International Conference on Mobile, Ubiquitous, and Intelligent Computing', MUSIC, 2012, 26–28 June 2012, Vancouver, BC.*

## 1 Introduction

Business processes consist of tasks which are performed to reach certain corporate goals (see e.g. Becker et al., 2000; Ouyang et al., 2009). In this context, the rising popularity of mobile computing technologies led to increasingly flexible business environments. Therefore, mobile technologies also have a strong impact on corresponding business processes, such as inventory management, sales-oriented processes, ambient assisted living support or processes in hospital environments (see e.g. Bardram et al., 2006; Consolvo et al., 2004; Hung et al., 2010; Roussos, 2006; Zhang et al., 2012). Many processes become more mobile, flexible and distributed and are executed in different contexts (Rosemann, 2008). As a result, mobile business processes may be executed at varying places, times, and by different people or devices (see e.g. Chakraborty and Lei, 2004; Giner et al., 2011; Gruhn et al., 2007).

In recent years, many organisations need to fulfil a number of compliance requirements stemming from certain laws and internal or external regulations. Thereby, IT compliance deals with the specification, implementation and maintenance of information systems that must comply with (new) regulatory requirements. Many compliance requirements covered in such regulations relate to business-level security concerns and thus directly affect an organisation's business processes. These process-related compliance requirements arise from, for example, the Sarbanes-Oxely Act (SOX), the Health Insurance Portability and Accountability Act (HIPPA) or the Basel II Accord (see e.g. Cannon and Byers, 2006; Damianides, 2004; Mishra and Weistroffer, 2007).

Yet, securing information systems is usually still based on documents without standardised structure and typically without alignment between business and IT perspectives (see e.g. Burn and Szeto, 2000; Chan et al., 2006; Klarl et al., 2009; Neubauer et al., 2006). In recent years, business processes are increasingly designed with security and compliance considerations in mind (see e.g. Bertino et al., 1999; Tan et al., 2004; Warner and Atluri, 2006; Wolter and Schaad, 2007). To protect sensitive data and services, context-aware security mechanisms have been identified as a prerequisite for secure mobile computing (see e.g. Leavitt, 2011; Miller, 2011; Shabtai et al., 2010; Stewart et al., 2007; Xiao et al., 2006). One important security aspects is *access control* which deals with the elicitation, specification, maintenance and enforcement of authorisation policies in software-based systems (see e.g. Sandhu and Samarati, 1994). In this paper, we focus on context-aware access control in a business process context.

In an IT-supported workflow, process-related *context constraints* are a means to consider context information in access control decisions (see e.g. Bertino et al., 2001; Georgiadis et al., 2001; Strembeck and Neumann, 2004). Typical examples for context constraints in organisational settings regard the temporal or spatial context of task execution, user-specific attributes, or the task execution history of a user (see e.g. Cuppens and Cuppens-Boulahia, 2008). Yet, standard process modelling languages, such as BPMN (OMG, 2011a) or UML Activity diagrams (OMG, 2011b), do not provide native language support to model process-related context constraints.

One objective of our research is to define process-related context constraints via native modelling language constructs. Usually, process models focus on the process-flow perspective and are decoupled from access control-relevant context information. In practice, the lack of native modelling constructs for context constraints results in a number of workarounds (Rosemann et al., 2008), for example: corresponding context constraints become part of the control flow by including several decision nodes, such as 'check if location is Vancouver' or 'check if user is registered'. As a result, process models become larger and more complex to read. Alternatively, multiple process models for different contextual scenarios are designed, leading to highly redundant models.

In recent years, Role-Based Access Control (RBAC; Ferraiolo et al., 2007; Sandhu et al., 1996) has developed into a de facto standard for access control in both, research and industry. In RBAC, roles correspond to different job-positions and scopes of duty within a particular organisation or information system (Strembeck, 2010). Access permissions are assigned to roles according to the tasks a role has to accomplish. Human users and other active entities are assigned to roles. Thereby, each subject acquires all permissions necessary to fulfil its duties via his/her role memberships. In addition, RBAC supports the definition of context constraints on various parts of an RBAC model (see e.g. Georgiadis et al., 2001; Strembeck and Neumann, 2004; Warner and Atluri, 2006).

In this paper, we integrate context constraints into process-related RBAC models (Strembeck and Mendling, 2011) and thereby support context-dependent task execution. We use a generic context concept where constraints can be defined for any type of context information that can be tracked in an information system. To achieve this, we formally embed RBAC context constraints into a business process context. Based on the formal model, we define a corresponding extension for a standard process modelling language. In particular, we define a UML extension for the integrated modelling of processes, RBAC concepts, and context constraints via extended UML2 activity diagrams. Our approach supports the complete and correct mapping of process definitions and related context-aware access control policies to corresponding software systems. This is essential to assure consistency between modelling-level specifications and software systems enforcing the respective access control policies. In addition to the modelling extension, an extension for a software platform was implemented to allow a direct mapping of the modelled processes including context constraints to corresponding source code structures (see Strembeck and Mendling, 2010; Strembeck and Mendling, 2011). The source code of our implementation is available for download (http://wi.wu.ac.at/home/mark/BusinessActivities/library.html). In comparison to the work of Schefer-Wenzl and Strembeck (2012), this paper provides an extended version of the formal metamodel, describes example processes modelled via our UML extension, introduces the software platform support, and offers an extended discussion of related work.

Section 2 introduces our formal metamodel for context-aware RBAC process models. The corresponding UML2 extension is defined in Section 3. Next, Section 4 discusses examples of business processes including context constraints. In Section 5, we present an overview of our software platform for context-aware RBAC process models. Finally, Section 6 discusses related work, and then Section 7 concludes the paper.

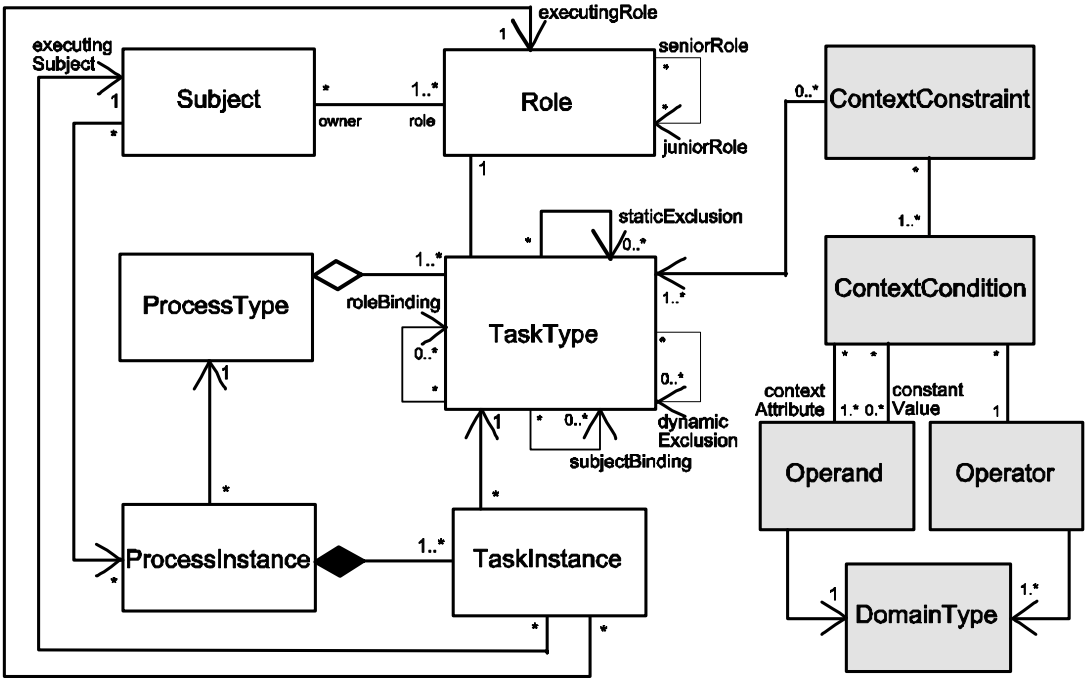
## 2 Formal metamodel for process-related context-aware RBAC models

Each *task* in a process (e.g. to sign a contract) is typically associated with certain access permissions (e.g. to read and write the contract document). Therefore, *subjects* participating in a workflow, i.e. human users or software-agents, must be authorised to perform the tasks needed to complete the process (see e.g. Georgiadis et al., 2001; Oh and Park, 2003). A *role* is an abstraction containing the tasks and associated permissions of a certain subject-type (Strembeck, 2005). The left-hand side of Figure 1 illustrates the essential relations between these elements in process-related RBAC models (see Strembeck and Mendling, 2011).

Tasks defined in business processes are always performed within a certain context. These contextual attributes, e.g. time, location or the executing subject, may influence access control decisions. Thus, depending on the context, different authorisation rules might apply for executing a particular task. A *context constraint* is a modelling-level concept defining that certain contextual attributes must meet certain predefined conditions to permit the execution of a specific task (Strembeck and Neumann, 2004). In particular, context constraints consist of context conditions that include context attributes and context functions: a *context attribute* represents a certain property of the environment whose actual value might change dynamically (e.g. time, date and location). For each context attribute, a *context function* exists that can obtain the current value of a specific context attribute (e.g. date() returns the current date). A *context condition* is a Boolean expression that restricts the permitted values of a context attribute (e.g. date > 01/01/2012).

In this paper, we focus on the modelling of context-aware RBAC process models. For this purpose, we formally embed context constraints into a business process modelling context. In Figure 1, the metamodel for process-related RBAC models (see Strembeck and Mendling, 2011) is extended with context constraints. A process-related context constraint is associated with a task and one or more context conditions. All context conditions must evaluate to true in order for the context constraint to be fulfilled. Each context condition consists of an operator, e.g. an infix operator, such as $=, \geq, >, <, \leq, \neq$, and two or more operands. One of the operands has to refer to a certain context attribute and the other operands are either context attributes or constant values (e.g. current location = Vancouver). Moreover, the type and range of values for each operand and for each operator is determined via its domain type (e.g. Boolean, date and integer). Definition 1 formally specifies the essential elements of the extended metamodel and their basic interrelations.

**Figure 1** Extended metamodel for context-aware RBAC models



Definition 1 *(context-aware business activity RBAC model): Let cBRM = (E,Q,D,CX) be a context-aware business activity RBAC model, where E refers to the pairwise disjoint sets of the metamodel, Q refers to mappings that establish relationships, D refers to binding and mutual exclusion constraints, and CX refers to mappings for context constraints*.

The *sets E of the context-aware business activity RBAC model* are:

- An element of *S* is called *Subject*. $S \neq \emptyset$
- An element of *R* is called *Role*. $R \neq \emptyset$ .
- An element of $P_T$ is called *Process Type*. $P_T \neq \emptyset$ .
- An element of $P_I$ is called *Process Instance*.
- An element of $T_T$ is called *Task Type*. $T_T \neq \emptyset$ .
- An element of $T_I$ is called *Task Instance*.
- An element of *CA* is called *Context Attribute*. $CA \neq \emptyset$ .
- An element of *CV* is called *Constant Value*. $CV \neq \emptyset$ .
- An element of *OD* is called *Operand*. $OD \neq \emptyset$ .
- An element of *DM* is called *Domain Type*. $DM \neq \emptyset$ .
- An element of *OT* is called *Operator*. $OT \neq \emptyset$ .
- An element of *CD* is called *Context Condition*.
- An element of *CC* is called *Context Constraint*.

For the mappings of the *business activity RBAC model* $(Q = rh \cup rsa \cup es \cup er \cup tra \cup ti, D = sb \cup rb \cup sme \cup dme)$ (see Strembeck and Mendling, 2011). Below, we define the additional *mappings for context constraints*: $CX = dm_{od} \cup dm_{ot} \cup od_{cd} \cup ot_{cd} \cup cond \cup fulfilled_{CD} \cup linked_{CD} \cup fulfilled_{CC} \cup linked_{CC}$ (*P* refers to the power set):

1 An operand is either a context attribute or a constant value: $\forall od \in OD : od \in CA \vee od \in CV$ .

2 Each operand has a certain domain type which determines the type and range of values for this operand (e.g. Boolean, date and integer). For example, the context attribute *currentDate* has the domain type *date*. Note that each operand has exactly one domain type: *The mapping* $dm_{od} : OD \mapsto DM$ is called *operand-domain type*. For $dm_{od}(od) = dm$ , we call $od \in OD$ operand and $dm \in DM$ the domain type specified for an operand.

3 In a context condition, an operator is applied to the operands. In contrast to operands, operators may be linked to multiple domain types. Consider, for example, the *lower-equal* (≤) operator, which may be used to compare numbers or dates, is linked to the domains: integer, real and date. Formally: the mapping $dm_{ot} : OT \mapsto P(DM)$ is called *operator-domain types*. For $dm_{ot}(ot) = DM_{ot}$ , we call $ot \in OT$ operator and $DM_{ot} \subseteq DM$ the set of domain types specified for an operator.

4 Context conditions are predicates that consist of operands and operators. Each context condition contains one operator and the number of operands required by this operator. In the example condition '*age* > 18', the context attribute age and the constant value 18 are the operands, whereas > is the operator.

- The mapping $od_{cd} : CD \mapsto P(OD)$ is called *context condition operands*. For $od_{cd}(cd) = OD_{cd}$ , we call $cd \in CD$ context condition and $OD_{cd} \subseteq OD$ is the set of operands included in this condition.
- The mapping $ot_{cd} : CD \mapsto OT$ is called *context condition operator*. For $ot_{cd}(cd) = ot$ , we call $cd \in CD$ context condition and $ot \in OT$ is the operator included in this condition.
- The mapping $cond : (P(OD) \times OT) \mapsto CD$ is called *context condition*. For $cond(OD_{cd}, ot) = cd$ , we call $OD_{cd} \subseteq OD$ the *set of operands included in this condition*, $ot \in OT$ is called the *operator*, and $ot \in OT$ is called *context condition*.

5 Within each context condition, all operands must have the same domain type: $\forall cd_x \in CD : \forall od_x, od_y \in od_{cd}(cd_x) : dm_{od}(od_x) = dm_{od}(od_y)$

6 Within each context condition, the domain type of the operands must correspond to the domain type of the operator. Otherwise, the operator cannot be applied on the operands: $\forall cd_x \in CD : \forall od_x \in od_{cd}(cd_x), ot \in ot_{cd}(cd_x) : dm_{od}(od_x) \in dm_{ot}(ot)$

7 A context condition must contain at least one context attribute: $\forall cd_x \in CD : \exists od_x \in od_{cd}(cd_x) : od_x \in CA$

8 Context conditions are Boolean expressions that are fulfilled if the operator applied to the corresponding operands evaluates to true: *The mapping* $fulfilled_{CD}$: $CD \mapsto BOOLEAN$ is called *context condition fulfilment*. For $fulfilled_{CD}(cd) = Boolean$ , we call $cd \in CD$ *context condition*. The mapping follows a two-valued logic returning exactly one truth value. Thus, the $fulfilled_{CD}$ mapping returns *true* if the operator applied on the corresponding operands evaluates to true. Otherwise, it returns *false*.

9 A context constraint is linked to one or more context conditions: *The mapping* $linked_{CD} : CC \mapsto P(CD)$ is called *context condition to constraint linkage*. For $linked_{CD}(cc) = CD_{CC}$ , we call $cc \in CC$ *context constraint* and $CD_{CC} \subseteq CD$ the set of conditions linked to this context constraint.

10 A context constraint is fulfilled if all linked conditions evaluate to true: *the mapping* $fulfilled_{CC} : CC \mapsto BOOLEAN$ is called *context constraint fulfilment*. For $fulfilled_{CC}(cc) = Boolean$ , we call $cc \in CC$ *context*

*constraint*. The mapping follows a two-valued logic returning exactly one truth value. Thus, the $fulfilled_{CC}$ mapping returns *true* if all conditions linked to the context constraint are true. Otherwise, it returns *false*. $fulfilled_{CC}(cc_x)=true \Leftrightarrow \forall cd_x \in linked_{CD}(cc_x): fulfilled_{CD}(cd_x)=true$

11 Context constraints are linked to task types: *the mapping* $linked_{CC}: T_T \mapsto P(CC)$ is called *context constraint to task linkage*. For $linked_{CC}(t)=CC_T$ we call $t \in T_T$ *constrained task* and $CC_T \subseteq CC$ the set of *context constraints linked to this task.*

For the purposes of the paper, Definition 2 repeats the specification of the process flow model for business activities as already defined by Strembeck and Mendling (2011). This formalisation defines the process flow model as a graph with specific types of nodes for actions and control elements, as well as arcs capturing the flow of control. It is specified as a labelled transition system (see e.g. Cortadella et al., 1998; Murata, 1989).

Definition 2 *(context-aware business activity process flow model): A process flow model* $PFM=(N,A)$ *where* $N=T_T \cup C_F \cup C_J \cup C_D \cup C_M \cup \{start,end\}$ *refers to pairwise disjoint sets and* $A \subseteq N \times N$.

*An element of N is called node and an element of A is called arc. Elements of* $T_T$ *are called task types. An element of* $C=C_F \cup C_J \cup C_D \cup C_M$ *is called control node. An element of* $C_F$ *is called fork, an element of* $C_J$ *is called join, an element of* $C_D$ *decision, and an element of* $C_M$ *is merge. start is called start node and end is called end node. All nodes* $n \in N$ *are on a path from start to end.*

Furthermore, before executing a task, we have to check, if it is possible to find a subject who is allowed to execute this task without violating associated context constraints, mutual exclusion constraints or binding constraints (Strembeck and Mendling, 2010). Therefore, we need to explicitly refer to the execution history of a process instance *p* to reflect which subject has already executed which task instance (see Strembeck and Mendling, 2011). Accordingly, we define an execution history *h*(*p*) for context-aware business activities in Definition 3.

Definition 3 *(execution history): Let* $cBRM=(E,Q,D,CX)$ *be a context-aware business activity RBAC model and* $P_I$ *its set of process instances. For a particular process instance* $p \in P_I$, *an execution event* $exec(p) \in (T_I \times T_T \times R \times S)$ *is a record of a particular task execution where* $T_I$ *refers to the set of task instances,* $T_T$ *refers to the set of corresponding task types, R refers to the set of executing roles, and S refers to the set of executing subjects. The execution history h(p) of a process instance p is defined as a mapping* $h: P_I \mapsto P(\{(t_x,t_t,r,s) \mid t_x \in T_I, t_t \in T_T, r \in R, s \in S\})$, *which maps h(p) to a set of execution events exec(p) (for further details, see Strembeck and Mendling 2011).*

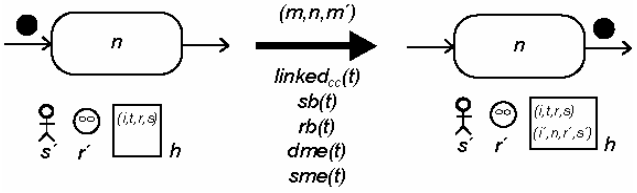
*If a particular task type* $t_t \in T_T$ *is linked to context constraint(s), i.e. if* $linked_{CC}(t_t) \neq \emptyset$, *the execution history includes a record of all context constraints linked to the task instances. For a particular task instance* $t_x$ *with* $t_x \in ti(t_t,p)$, *the corresponding task type, each context constraint cc linked to this task, and the results of evaluating the context constraint, i.e.* $fulfilled_{CC}(cc)=true$ *or* $fulfilled_{CC}(cc)=false$, *are documented. Thus, for each linked context constraint, i.e.* $\forall cc \in linked(t_t)$, *the context-aware execution history* $h_c(t_x)$ *of a task instance* $t_x$ *is defined as a mapping* $h_c: T_I \mapsto P(\{(cc,result) \mid cc \in CC\})$ *with* $result=fulfilled_{CC}(cc)$.

Definition 4 specifies a notion of state for context-aware business activity models as already defined by Strembeck and Mendling (2011). It is based on the distribution of control tokens on the arcs and the execution history. The letter *m* refers to a state and *M* refers to the set of states, respectively. The letter *M* is derived from the term *marking* used in Petri nets.

Definition 4 *(state of a process instance): Let PFM be a context-aware business activity process flow model and cBRM a context-aware business activity RBAC model such that the tasks of the first match the latter, that is* $T_T^{PFM}=T_T^{BRM}$, *and such that all* $a \in A$ *are the arcs of PFM. The state of a process instance* $p \in P_I$ *is defined as a pair* $m=(d,h)$ *where h refers to the execution history of p and where d is an element of* $D: A \mapsto N$, *which is a distribution of tokens on the arcs of the model with N being the natural numbers. The initial state* $m_i=(d_i,h)$ *of p is a state m such that* $h=\emptyset$ *and for each* $a=(n1,n2)$ *with* $n1=start: d(a)=1$ *and for each* $a=(n1,n2)$ *with* $n1 \neq start: d(a)=0$.

Figure 2 shows the transition relations for task nodes depicted via round-cornered rectangles. In a transition from one task to another task, different kinds of constraints defined on these tasks are considered. Hereby, $linked_{cc}$ refers to context constraints linked to a task. Moreover, dynamic and static mutual exclusion constraints (short: *dme*, *sme*) as well as subject-binding and role-binding constraints (short: *sb*, *rb*) are considered. A detailed discussion on mutual-exclusion and binding constraints was found in the work of Strembeck and Mendling (2011).

**Figure 2** Transition for task nodes in *ContextAwareBusinessActivities*




Before a transition to a subsequent task is performed, we have to consider the availability of subjects with suitable roles, such that, given the execution history, none of the context constraints, mutual exclusion constraints, and

binding constraints are violated. Then, the state change can be applied and a new execution entry is added to the history. If there is no subject-role combination that is allowed to execute a task at a particular point in time, then the process instance will deadlock. In Definition 5, we refer to the set of incoming and outgoing arcs of a node as follows: for each node, $n \in N$, we define the set of incoming arcs $n_{in} = \{(x,n) | x \in N \wedge (x,n) \in A\}$, and the set of outgoing arcs $n_{out} = \{(n,y) | y \in N \wedge (n,y) \in A\}$. In addition, the mapping $ti(n,p)$ with $n \in T_T$ and $p \in P_I$ refers to the set of task instances in a particular process instance (Strembeck and Mendling, 2011). The respective reachability graph definition in terms of a labelled transition system (see e.g. Cortadella et al., 1998; Murata, 1989) is given in Definition 5.

Definition 5 *(Reachability Graph): Let* $PFM = (N, A) n \in N$ *be a business activity process flow model, cBRM a context-aware business activity RBAC model including the set of task instances* $T_I$*, a process instance* $p \in P_I$*, and the execution history* $h$ *of* $p$*. Then the reachability graph* $RG = (M, TR)$ *with* $TR \subseteq M \times N \times M$ *must comply with the following constraints*:

- The initial state is in M, i.e. $m_i \in M$ with $T_I = \emptyset$.
- If $m = (d,h) \in M$ and $n \in T_T$ and for all $a \in n_{in} : d(a) > 0$ and there exists $i' \in ti(n,p), r' \in R, s' \in S$ with $n \in tra(r')$ and $r' \in rsa(s')$ such that for all $(i', n, r', s') \in h$ holds that:

*If all context constraints linked to a task are fulfilled:*

if $linked_{CC}(n) \neq \emptyset$ and $\forall i' \in ti(n,p) : \forall cc \in linked_{CC}(n)$: $fulfilled_{CC}(cc) = true$ and

$m' = (d', h')$ exists such that

$\forall a \in A \setminus (n_{in} \cup n_{out}) : d'(a) = d(a)$,

*there exists at least one* $a \in n_{in} : d'(a) = d(a) - 1$*, and*

*there exists at least one* $a_{fulfilled} \in n_{out} : d'(a) = d(a) + 1$,

*and* $h' = h \cup \{(i', n, r', s')\}, T_I' = T_I \cup \{i'\}$ *with*

$i' \in ti(n,p), es(i') = s'$*, and* $er(i') = r'$*, and* $m' \in M$

*and* $(m, n, m') \in TR$.

*If a context constraint linked to a task is not fulfilled:*

*if* $linked_{CC}(n) \neq \emptyset$ *and*

$\exists i' \in ti(n,p) : \exists cc \in linked_{CC}(n) : fulfilled_{CC}(cc) = false$ *and*

$m' = (d', h')$ *exists such that*

$\forall a \in A \setminus (n_{in} \cup n_{out}) : d'(a) = d(a)$,

*there exists at least one* $a \in n_{in} : d'(a) = d(a) - 1$*, and*

*there exists at least one* $a_{notfulfilled} \in n_{out} : d'(a) = d(a) + 1$,

*and* $h' = h \cup \{(i', n, r', s')\}, T_I' = T_I \cup \{i'\}$ *with*

$i' \in ti(n,p), es(i') = s'$*, and* $er(i') = r'$*, and* $m' \in M$

*and* $(m, n, m') \in TR$.

Thus, if a context constraint is not fulfilled, an alternative path can be defined, leading, for example, to the end of a process.

## 3 Modelling context constraints in UML

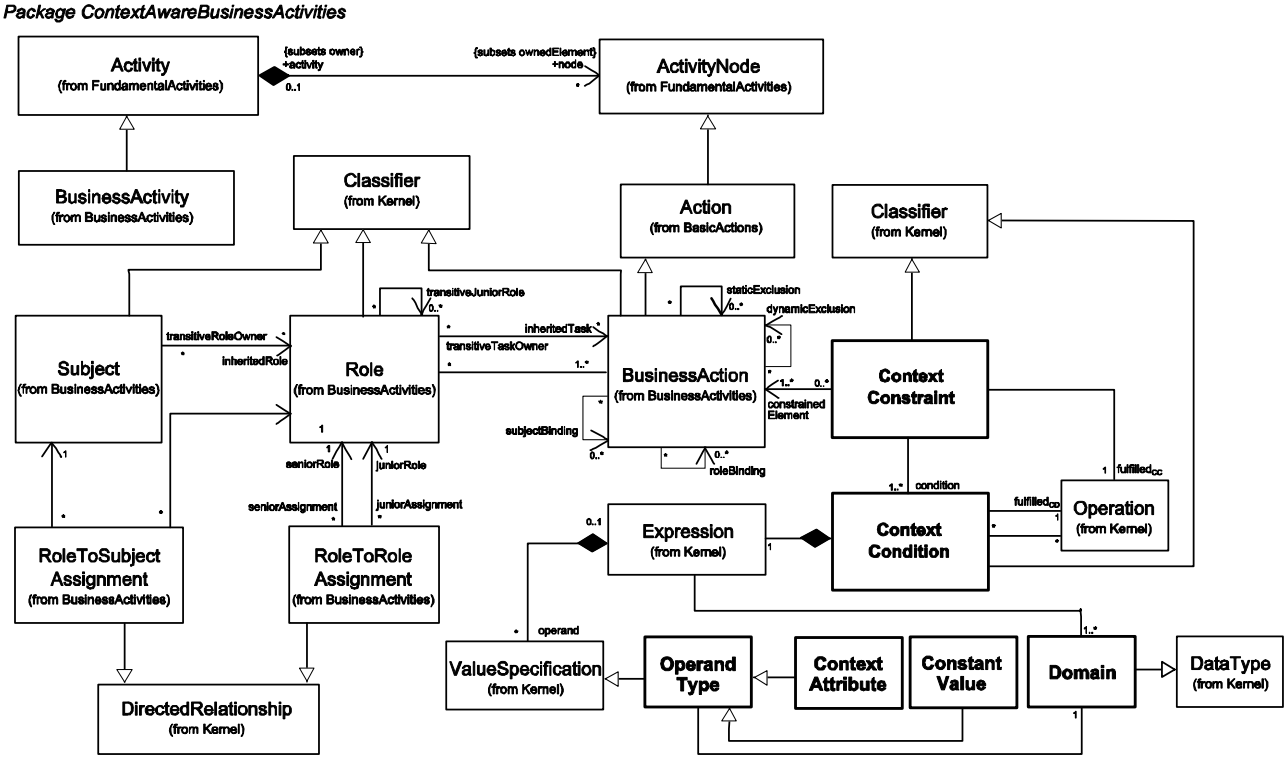
The Unified Modelling Language (UML) (OMG, 2011b) is the de facto standard for the specification of information systems. Modelling support for context constraints via a standard notation can help to bridge the communication gap between software engineers, security experts, experts of the application domain, and other stakeholders (see e.g. Mouratidis and Jürjens, 2010). Our modelling approach for context-aware access control concepts acts as an enabler to document and communicate how access control in general and context constraints in particular affect a business process.

UML2 Activity models offer a process modelling language that allows us to model the control and object flows between different actions. The main element of an activity diagram is activity. Its behaviour is defined by a decomposition into different actions. A UML2 activity thus models a process while actions included in the activity are used to model tasks (for details on UML2 activity models; see OMG, 2011b).

In addition, we use the Object Constraint Language (OCL) (OMG, 2010) to formally define the semantics of the newly introduced UML elements and to ensure the consistency of the extended UML models. Corresponding software tools can enforce OCL invariants on the modelling-level as well as in runtime models. Thereby, we can ensure the consistency of our extended UML models with the definitions provided above. However, note that our general approach does not depend on the UML and may also be applied to extend other process modelling languages.

The UML standard provides two options to adapt its metamodel to a specific area of application (OMG, 2011b): (a) defining a UML profile specification, which does not change the UML metamodel but extend existing UML meta-classes for special domains; (b) extending a UML metamodel, which allows for the definition of new elements with customised semantics. However, UML profiles are not a first-class extension mechanism (see OMG, 2011b, p.660). Moreover, the newly defined modelling elements for context constraints require new semantics which are not available in the UML metamodel. Thus, we introduce the UML metamodel extension *ContextAwareBusinessActivities* for modelling process-related context constraints (see Figure 3). Our UML extension extends the *BusinessActivities* package (Strembeck and Mendling, 2011), which provides UML modelling support for process-related RBAC models.

**Figure 3** UML2 metamodel extension *ContextAwareBusinessActivities*




A *BusinessActivity* (Strembeck and Mendling, 2011b) is a special UML activity (see Figure 3). A *BusinessAction* corresponds to a task and comprises all permissions to perform the task. *Roles* and *SUBJECTS* are linked to BusinessActions directly or transitively. Furthermore, mutual exclusion and binding constraints can be defined on BusinessActions (see Strembeck and Mendling, 2011b, for further details).

For integrating context constraints into process-related RBAC models according to the definitions provided in Section 2, we introduce the following new meta-classes: The *ContextConstraint* meta-class is a special UML2 classifier (from the Kernel package, see OMG, 2011b and Figure 3). Each ContextConstraint is linked to one or more BusinessActions and one or more ContextConditions indicating that the constrained BusinessAction can only be performed if all conditions specified in the ContextConstraint are fulfilled (see OCL Constraint 1 listed at the end of this section).

In UML, each classifier may include an arbitrary number of operations (see OMG, 2011b). A ContextConstraint defines one mandatory operation called $fulfilled_{cc}$. For each ContextConstraint, the corresponding $fulfilled_{cc}$ operation checks if all linked ContextConditions are fulfilled and returns either true or false (see Constraint 5).

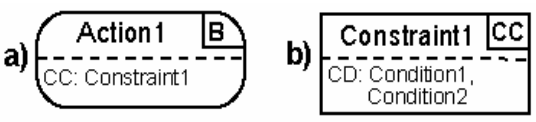
A *ContextCondition* is a special type of classifier. Moreover, each ContextCondition contains one UML expression. According to OMG (2011b), each expression can define a set of operands of the type ValueSpecification. An expression thereby represents an operator which is applied to those operands, e.g. $\geq(size, 5)$ . A ContextCondition can include operands of the types *ContextAttribute* (e.g. age, size and date) or *ConstantValue* (e.g. constant integer numbers or strings) (see Constraint 2). At least one of the operands included in a ContextCondition must be a ContextAttribute (see Constraint 3).

In addition, each ContextCondition includes one mandatory operation called $fulfilled_{CD}$ and an arbitrary number of other operations. For each ContextCondition, the corresponding $fulfilled_{CD}$ operation checks if the context condition is fulfilled and returns either true or false (see Constraint 6).

A *Domain* is a subtype of the UML2 DataType meta-class (see OMG, 2011b). Operands are linked to a certain domain. Similarly, the operator defined in the expression of a ContextCondition is linked to one or more domains (see Figure 3). Within the same ContextCondition, the domain of the operands needs to correspond to the domain of the operator (see Constraint 4). Otherwise, a ContextCondition cannot be evaluated.

Figure 4 shows presentation options to visualise the relations between BusinessActions and ContextConstraints as well as between ContextConstraints and ContextConditions. Note that these relations are formally defined through our UML metamodel extension and therefore exist independent of their actual graphical representation.

**Figure 4** Visualising (a) context-constrained BusinessActions and (b) ContextConstraints




Constraint 1: *A ContextConstraint defines that certain conditions must be met in order to be able to execute a*

*particular BusinessAction. Thus, a ContextConstraint refers to a BusinessAction and is specified via one or more ContextConditions*:

```
context ContextConstraint
inv: self.constrainedElement->forAll(c |
       c.oclIsKindOf(BusinessAction))
inv: self.condition->forAll(cd |
       cd.oclIsKindOf(ContextCondition))
```

Constraint 2: *Each ContextCondition consists of an operator and operands. The operands specified in a ContextCondition are either ContextAttributes (e.g. location) or ConstantValues (e.g. Vienna)*:

```
context ContextCondition
inv: self.expression.operand.
     oclAsType(OperandType)->forAll(o |
        o.oclIsKindOf(ContextAttribute) or
        o.oclIsKindOf(ConstantValue))
```

Constraint 3: *In order to define a valid ContextConition, at least one operand in each ContextCondition needs to be a ContextAttribute*:

```
context ContextCondition
inv: self.expression.operand.
     oclAsType(OperandType)->exists(o |
        o.oclIsKindOf(ContextAttribute))
```

Constraint 4: *Within the same ContextCondition, all operands must have the same domain which determines the type and range of values this operand may take (e.g. date). Moreover, the operands' domain has to correspond to one of the operator's domains*:

```
context ContextCondition
inv: self.expression.operand->forAll(od1, od2 |
       od1.oclAsType(OperandType).domain.name =
       od2.oclAsType(OperandType).domain.name and
       self.expression.domain->exists(d |
         d.name =
         od1.oclAsType(OperandType).domain.name))
```

Moreover, the following two constraints must be satisfied which cannot be expressed in OCL (see OMG, 2011b):

Constraint 5: *The $fulfilled_{CC}$ operations must evaluate to true to fulfil the corresponding ContextConstraint*.

Constraint 6: *The $fulfilled_{CD}$ operations must evaluate to true to fulfil the corresponding ContextCondition*.

Table 1 gives an overview of how each of the generic definitions in Section 2 is mapped to our UML extension for context-aware business activities.

**Table 1** Consistency between generic metamodel and UML extension

| *Generic definition* | *Covered through* |
|---|---|
| Definition 1: sets of the metamodel | Defined via new meta-classes in our metamodel extension |
| Definition 1.1: $\forall od \in OD: od \in CA \vee od \in CV$ | Constraint 2 |
| Definition 1.1: $dm_{od}: OD \mapsto DM$ | Defined via the *domain* association in our metamodel extension |
| Definition 1.1: $dm_{ot}: OT \mapsto P(DM)$ | Defined via the *domain* association in our metamodel extension |
| Definition 1.1: $od_{cd}: CD \mapsto P(OD)$ | Defined via the *ContextCondition* meta class and the *operand* association |
| Definition 1.1: $ot_{cd}: CD \mapsto OT$ | Defined via the *ContextCondition* and Expression meta class |
| Definition 1.1: $cond: (P(OD) \times OT) \mapsto CD$ | Defined via the *ContextCondition* meta class |
| Definition 1.1: $\forall cd_x \in CD: \forall od_x, od_y \in od_{cd}(cd_x): dm_{od}(od_x) = dm_{od}(od_y)$ | Constraint 4 |
| Definition 1.1: $\forall cd_x \in CD: \forall od_x \in od_{cd}(cd_x): dm_{od}(od_x) \in dm_{ot}(ot)$ | Constraint 4 |
| Definition 1.1: $\forall cd_x \in CD: \exists od_x \in od_{cd}(cd_x): od_x \in CA$ | Constraint 3 |
| Definition 1.1: $fulfilled_{CD}: CD \mapsto BOOLEAN$ | Constraint 6 |
| Definition 1.1: $linked_{CD}: CC \mapsto P(CD)$ | Defined via the *condition* association and Constraint 1 |
| Definition 1.1: $fulfilled_{CC}: CC \mapsto BOOLEAN$ and $\forall cd_x \in linked_{CD}(cc_x): fulfilled_{CD}(cd_x) = true$ | Constraints 5, 6 |
| Definition 1.1: $linked_{CC}: T_T \mapsto P(CC)$ | Defined via the *constrainedElement* association and Constraint 1 |
| Definition 2: $PFM = (N, A)$ | Implicitly defined via our metamodel extension and the specification of UML Activity models (see Figure 3 and OMG, 2011b) |
| Definition 3: $h: P_I \mapsto P(\{(t_i, t_t, r, s) \mid t_i \in T_I, t_t \in T_T, r \in R, s \in S\})$ | Instantiation of metamodel extensions (see OMG, 2011b) |
| Definition 4 $m = (d, h)$ | Implicitly defined via our metamodel extension and the specification of UML Activity models (see Figure 3 and OMG, 2011b) |
| Definition 5: reachability graph | Defined via the *constrainedElement* association and Constraints 1, 5, 6 |

## 4 Examples for context-aware RBAC models

In this section, we show two examples of how our UML extension for context-aware business activities can be used in different domains to define context-aware RBAC models for business processes.

### 4.1 A mobile inventory management process

Figure 5a shows the control-flow of a mobile inventory management process modelled as a context-aware business activity. We use the so-called UML swimlane notation (OMG, 2011b) to group actions which are executed by members of the same role. In this process, a retail store has implemented a mobile RFID solution, where RFID labels are placed on the products and sales representatives are outfitted with RFID handheld readers. These handhelds can communicate with back-end billing and inventory management applications. The process starts when a sales representative visits the customer's retail store where the consigned or vendor-managed inventory resides. The sales representative uses the handheld to *read the RFID tagged inventory* in the store. The current *inventory level is compared* with, e.g., the last inventory level to identify the amounts of items which have been removed. The *order information is sent* to back-end applications. If more than 100 items are ordered, the retail store manager needs to *approve the orders*. Next, the sales representative *generates an invoice*. Finally, the *order history is updated* via the back-end applications.

In this process, two BusinessActions are associated with context constraints (see Figure 5b). Before performing the *read RFID tagged inventory* task, the inventory management server has to check the context constraint *read RFID tags* including the following context conditions: First, the *sales_rep_id* context attribute must correspond to a *registered_sales_rep_id* to ensure that the corresponding sales representative is authorised to read the data for this store. Second, the *communication_protocol* context attribute has to be an encrypted protocol to ensure the secure transmission of potentially sensitive data. In addition, the *send orders to back-end* task is associated with the context constraint *send RFID orders* stating that the *client_MAC_address* context attribute must be registered to ensure that only registered PCs can be used to access the back-end applications. If the PC is not registered, the sales representative can request for authentication in order to be able to successfully send the orders to the back-end applications. Moreover, the *communication_protocol* must also be encrypted for secure data transmission.

### 4.2 An online examination process

Figure 6a shows an online examination process modelled as context-aware business activity. The process starts when a lecturer *uploads* an online exam document. If the exam is approved, a student can send a *fetch* request to receive the exam document. After the student's fetch request, the exam-server then *generates* an individual exam scrambling for the student and *sends* this exam to the student. If the sending process for the exam document is successful, the student performs the *do examination* task. In case an error occurs when sending the exam document, the student has to send another fetch request. After performing the examination, the student finally *dispatches* the completed exam document back to the server.

**Figure 5** Mobile inventory management process including context constraints

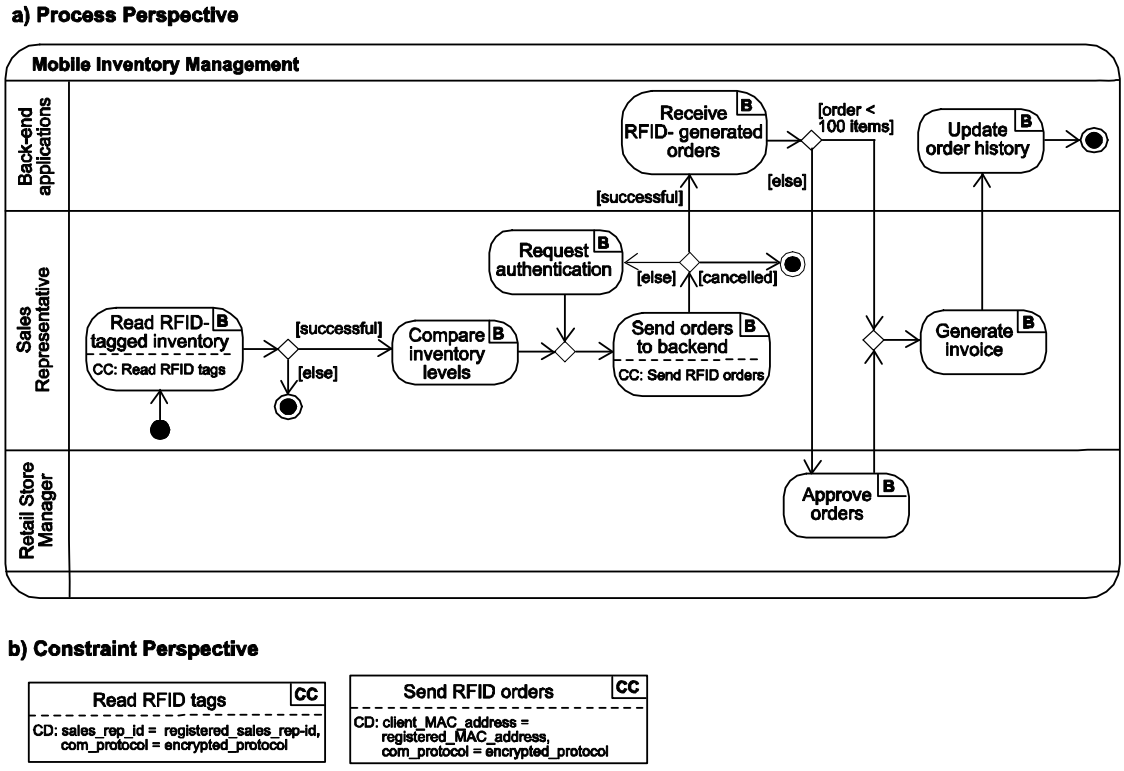
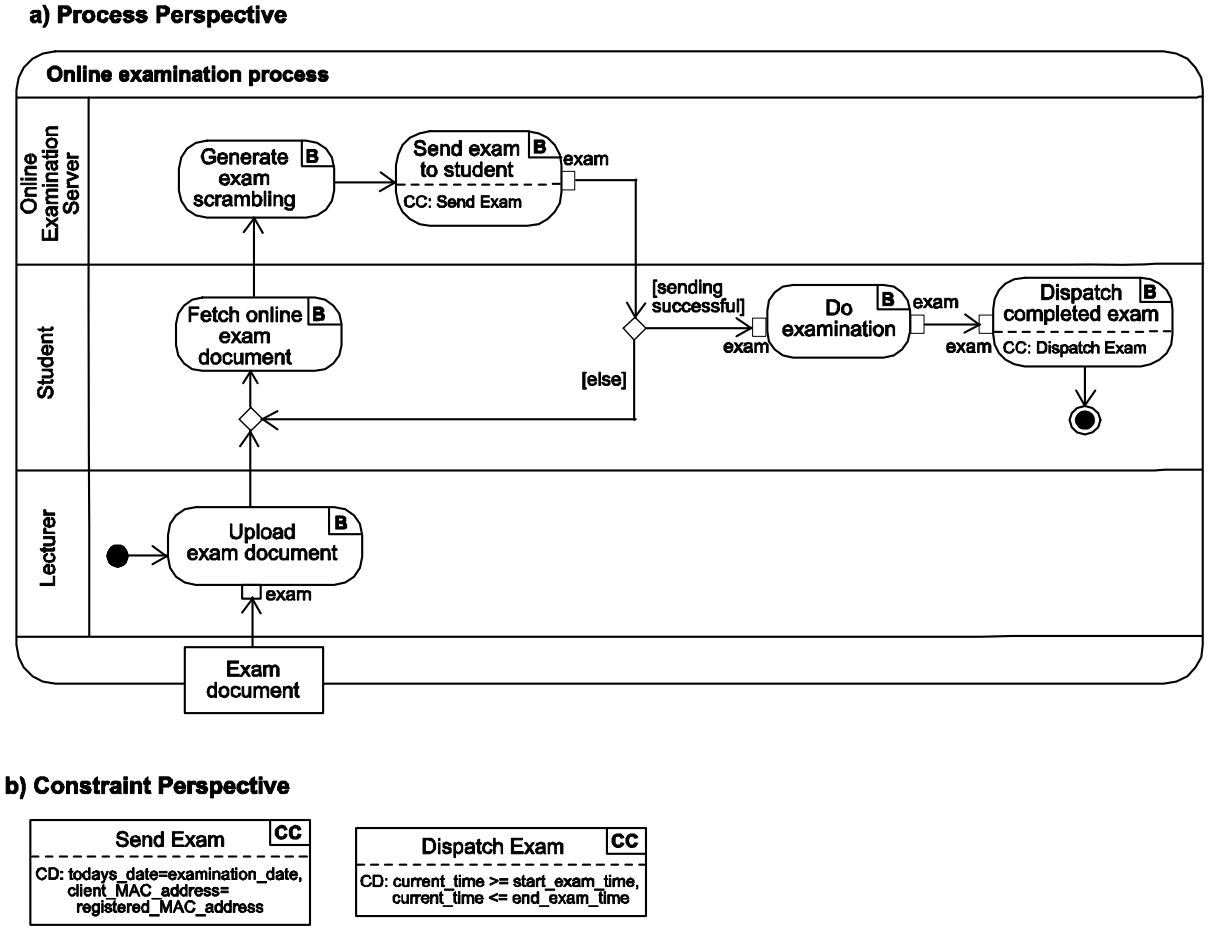


**Figure 6** Online examination process including context constraints




In this process, two BusinessActions are associated with context constraints (see Figure 6b). Before performing the task *send exam document to student*, the exam-server has to check the context constraint *send exam* including the following context conditions: First, the *todays_date* context attribute must correspond to the actual *examination_date*. Second, the *client_MAC_address* context attribute must be registered to ensure that only registered PCs can be used to access the exam-server. In addition, the *dispatch completed exam* task is associated with the context constraint *dispatch exam* stating that a student can only hand in the exam within the specified examination time. Thus, the *current_time* context attribute must be within the specified start and end examination time.

## 5 Platform support

In this section, we provide an overview of our platform support for context-aware RBAC process models. First, we present the Business Activity library and runtime engine. For the purposes of this paper, we extended this library with the *ContextConstraints* package (available at http://wi.wu.ac.at/home/mark/BusinessActivities/library.html).

The Business Activity library and runtime engine is a software platform that can manage process-related RBAC runtime models and enforce access control policies as well as several kinds of entailment constraints (Strembeck and Mendling, 2011). It supports all modelling level artefacts of process-related RBAC models and provides functions for managing corresponding runtime instances. Moreover, it automatically enforces all invariants defined via OCL constraints. Figure 7 shows an excerpt of the essential class relations of the Business Activity library and runtime engine. In particular, the Business Activity library and runtime engine provides the following basic features:

- Definition of subjects, roles and business actions;
- Specification of role-hierarchies;
- Definition of static (SME) and Dynamic Mutual Exclusion (DME) constraints as well as role- and subject-binding constraints for tasks;
- Assignment of tasks to roles (task-to-role assignment) and of roles to subjects (role-to-subject assignment);
- Revocation of task-to-role assignment and role-to-subject assignment relations;
- Functions for checking SME and DME constraints as well as role- and subject-binding constraints at runtime;
- Task allocation mechanism that ensures the consistency of all process instances at runtime;
- Several introspection functions for business actions, roles and subjects.

The ContextConstraints package extends the Business Activity library and runtime engine with support for context-aware RBAC process models as defined in the previous sections. Figure 8 shows the essential class relations of the ContextConstraints extension package. The Business Activity library and runtime engine as well as the ContextConstraints extension package are implemented via the programming language eXtended Object Tcl (XOTcl, see e.g. Neumann and Sobernig, 2009; Neumann and Sobernig, 2011; Neumann and Zdun, 2000). XOTcl is an object-oriented extension of the scripting language Tcl (Ousterhout, 1990) and is publicly available in website: http://www.xotcl.org/. XOTcl is a C-library that can be dynamically loaded into Tcl-compatible environments and is embeddable into C programs. Amongst others, XOTcl provides a mixin mechanism (Zdun et al., 2007). XOTcl mixin classes are a dynamic message interception technique. They allow us to define extension classes in addition to the inheritance hierarchy.

XOTcl supports per-object mixins as well as per-class mixins. Per-object mixins are classes that are applied as mixins for an individual instance of a class while per-class mixins are classes that are applied as mixins for a class (see Zdun et al., 2007, for details). Both XOTcl mixin constructs are used in the ContextConstraints extensions package to dynamically activate or deactivate certain behaviour for a class or object. For instance, if a task, which is associated with one or more context constraints, is allocated to a certain subject, the task-allocation function of the ContextConstraints package is invoked via a per-class mixin (to check if all linked context constraints are fulfilled before executing the next task) instead of the original task-allocation function provided by the Business Activity library and runtime engine. The ContextConstraints package provides the following basic features:

- Definition of context constraints and context conditions;
- Linking and unlinking context constraints and context conditions;
- Linking and unlinking tasks and context constraints;
- Specification of operands and operators for particular context conditions;
- Evaluation of context constraints for all tasks in a certain process instance at runtime;
- Redefinition of the task allocation mechanism to ensure that all associated context constraints are fulfilled at runtime (e.g. tasks may only be allowed to be performed at a certain time or date);
- Several introspection functions for context constraints, context conditions and additional introspection functions for business actions.

The Business Activity library and runtime engine in combination with the ContextConstraints extension package ensures the compliance of processes modelled via the *ContextAwareBusinessActivities* and user-defined context constraints. Thereby, it supports a straightforward mapping of modelling level context-aware RBAC models to the corresponding runtime models.

**Figure 7** Class model of the Business Activity library and runtime engine (Strembeck and Mendling, 2011)

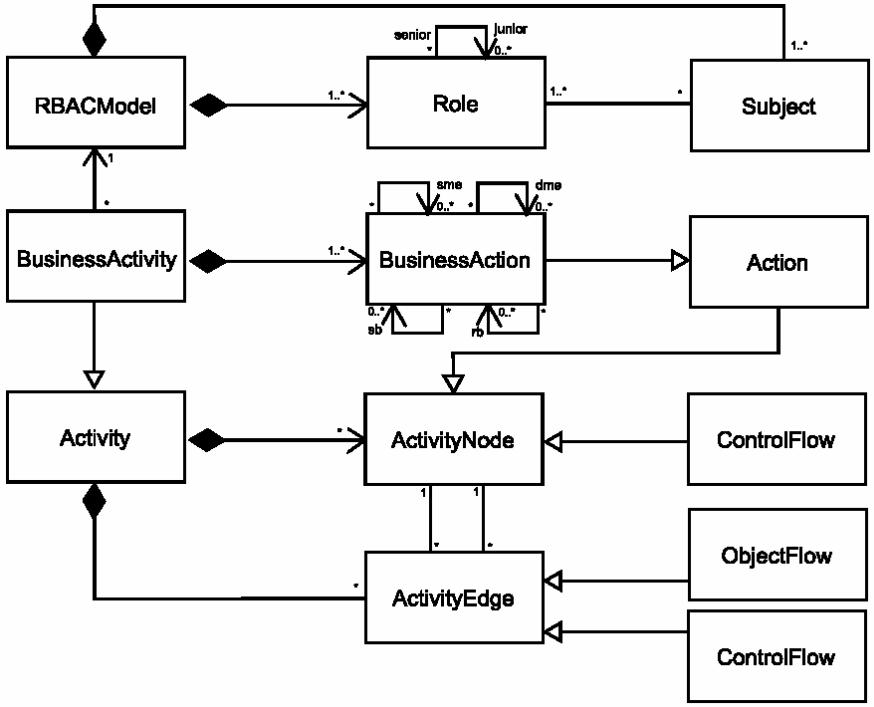
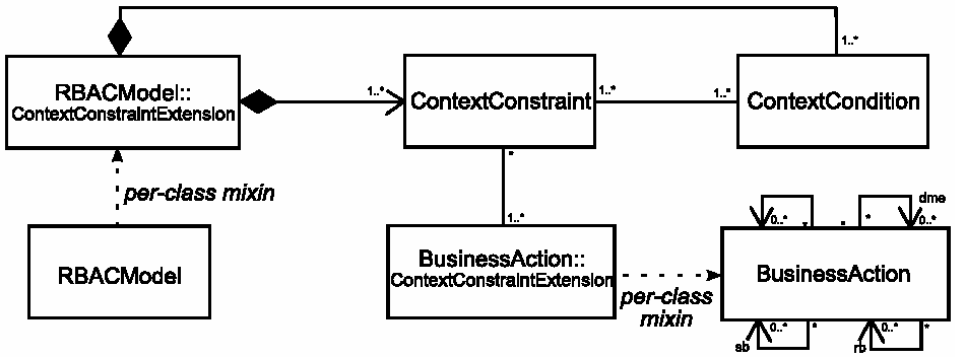


**Figure 8** Class model of the ContextConstraint package




## 6 Related work

In general, we distinguish three types of related work for the paper. First, we have approaches that primarily aim to integrate different types of context information into (role-based) access control models. Second, a number of different approaches exist that integrate context constraint information into a business process/workflow environment. We see these two categories as related work in the narrower sense. Third, some UML-based modelling approaches include context information for various other domains. Thus, we see these approaches as related work in a broader sense. Moreover, many of the RBAC and business process-related approaches are complementary to our work and are well-suited to be combined with our context-aware RBAC models.

Table 2 shows an overview of related work on modelling context constraints in an RBAC or business process context. With respect to the concepts and artefacts specified in Sections 2 and 3, we use a ✓ if a related approach provides similar and/or comparable support for a certain concept, and a Δ if a related approach provides at least partial support for a particular aspect.

**Table 2** Comparison of related work

| | *Business Processes* | *Roles* | *Arbitrary Context* | *Formal metamodel* | *Modelling support* | *Consistency checks* | *Tool support* |
|---|---|---|---|---|---|---|---|
| *Approaches to integrate context constraints into (role-based) access control* | | | | | | | |
| Temporal RBAC (Bertino et al., 2001) | – | ✓ | – | ✓ | – | ✓ | ✓ |
| Generalised TRBAC (Joshi et al., 2005) | – | ✓ | – | ✓ | – | ✓ | ✓ |
| GEO-RBAC (Bertino et al., 2005) | – | Δ | – | ✓ | – | ✓ | – |
| UbiCOSM (Corradi et al., 2004) | – | Δ | – | – | – | – | ✓ |
| RBAC in ambient space (Wedde and Lischka, 2004) | – | ✓ | – | ✓ | – | – | – |
| Team-based access control (Thomas, 1997; Georgiadis et al., 2001) | – | ✓ | – | ✓ | – | – | ✓ |
| CA-RBAC (Kulkarni and Tripathi, 2008) | – | ✓ | – | – | – | Δ | Δ |
| *Approaches to integrate context constraints into workflow environments* | | | | | | | |
| Workflow authorisation model (Adam et al., 1998; Atluri and Huang, 1996) | ✓ | Δ | – | ✓ | – | Δ | Δ |
| Context-sensitive access control (Cholewka, 2000) | ✓ | ✓ | – | – | – | – | Δ |
| Contextual security policies (Cuppens and Cuppens-Boulahia, 2008) | ✓ | Δ | – | ✓ | – | – | – |
| *UML-based modelling approaches* | | | | | | | |
| Context modelling profile (Simons, 2007) | ✓ | Δ | Δ | – | Δ | Δ | – |
| ContextUML (Sheng and Benatallah, 2005) | – | – | Δ | – | Δ | – | – |
| Privacy-aware Context Profile (Kapitsaki and Venieris, 2008) | – | Δ | Δ | – | Δ | – | – |
| Context-aware RBAC models (our approach) | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

Several approaches formally *integrate different types of context constraints into (role-based) access control models*. The RBAC model was extended to consider temporal aspects in access control decision (see e.g. Bertino et al., 2001; Joshi et al., 2005) or the users' location (see e.g. Bertino et al., 2005; Corradi et al., 2004; Wedde and Lischka, 2004). In the work of Georgiadis et al. (2001), the integration of contextual information with Team-based Access Control (TMAC) is discussed. Hereby, users obtain permissions according to their membership in roles and teams. Kulkarni and Tripathi (2008) presented a context-aware RBAC model, where context constraints are defined on different parts of the model. Moreover, role revocation is supported, in case values of the user attributes no longer satisfy the constraints. These approaches usually focus on the integration of certain contextual information, such as time, location or user attributes, into the RBAC model. In our approach, context constraints can be defined for any contextual information tracked in an information system.

The notion of *context constraints in workflow environments* has been studied, e.g. in the Workflow Authorisation Model (WAM) (Adam et al., 1998; Atluri and Huang, 1996). In this approach, subjects only can access objects during the execution of a task. For this purpose, temporal constraints are defined on tasks. A similar approach was presented by Cholewka et al. (2000), where permissions in workflows are granted according to the actual task. Another approach for contextual security policies in organisational environments was presented by Cuppens and Cuppens-Boulahia (2008), where context is viewed as an extra condition that must be satisfied to activate a security rule. However, only certain types of contextual attributes, such as time or location are considered in all of these approaches. Moreover, they do not offer modelling support for context-aware access control concepts in business processes.

Other approaches for modelling context exist for various domains. There are some *UML-based modelling approaches* for considering context constraints in mobile or distributed systems. In contrast to our modelling approach, none of these works considers the business process/workflow perspective. Simons (2007) presented a UML profile for modelling context for mobile distributed systems via special UML class diagrams. Similar to our approach, a generic context concept is used which is not limited to a certain type of context information. Moreover, OCL constraints define additional consistency checks for the extension. In contrast to our approach, there is no definition of a formal metamodel. Thus, the approach is limited to the UML. Moreover, the process flow cannot be visualised and tool support for the CMP profile is missing. The ContextUML language (Sheng and Benatallah, 2005) is another UML profile for modelling context-aware web-services. ContextUML does not specify access control mechanisms. Moreover, a generic metamodel and consistency checks are not specified. Kapitsaki and Venieris (2008) presented a similar approach. The authors present the Privacy-aware Context Profile (PCP) in UML notation considering several account privacy and quality aspects. A formal representation as well as consistency checks and tool support are not provided for this UML profile.

To the best of our knowledge, this paper presents the first attempt to address RBAC context constraints from a business process modelling perspective. While many sophisticated approaches exist that formally integrate process-related context constraints into access control models, corresponding process modelling support is largely missing. Our approach complements existing context modelling approaches by providing modelling support for business processes and corresponding contextual authorisation constraints in a consolidated modelling language.

## 7 Conclusion

This paper was motivated by the need for considering context-aware access control mechanisms in business processes. This is especially important due to the rising importance of mobile computing technologies in business environments. We defined a formal metamodel to integrate context constraints into process-related RBAC models. Based on these definitions, we extended the UML to allow for the model-based specification of context-aware RBAC process models in UML activity diagrams. Moreover, we apply the OCL to define the semantics of the newly introduced UML meta-classes. Our extension can be integrated with other UML-based approaches or tools. We also implemented a context-constraint extension for the BusinessActivity library and runtime engine.

## References

Adam, N.R., Atluri, V. and Huang, W-K. (1998) 'Modeling and analysis of workflows using Petri nets', *Journal of Intelligent Information Systems*, Vol. 10, No. 2.

Atluri, V. and Huang, W-K. (1996) 'An authorization model for workflows', *Proceedings of European Symposium on Research in Computer Security*, Vol. 1146, pp.44–64.

Bardram, J.E., Hansen, T.R., Mogensen, M. and Soegaard, M. (2006) 'Experiences from real-world deployment of context-aware technologies in a hospital environment', *Proceedings of the 8th International Conference on Ubiquitous Computing (UbiComp)*, Vol. 4206, pp.369–386.

Becker, J., Rosemann, M. and Uthmann, C.V. (2000) 'Guidelines of business process modeling', *Proceedings of Business Process Management, Models, Techniques, and Empirical Studies*, Vol. 1806, London, UK, pp.30–49.

Bertino, E., Bonatti, P.A. and Ferrari, E. (2001) 'TRBAC: A Temporal Role-based Access Control model', *ACM Transactions on Information and System Security (TISSEC)*, Vol. 4, No. 3, pp.191–233.

Bertino, E., Catania, B., Damiani, M.L. and Perlasca, P. (2005) 'GEORBAC: A spatially aware RBAC', *Proceedings of the 10th ACM Symposium on Access Control Models and Technologies (SACMAT)*, NY, USA, pp.29–37.

Bertino, E., Ferrari, E. and Atluri, V. (1999) 'The specification and enforcement of authorization constraints in workflow management systems', *ACM Transactions on Information and System Security (TISSEC)*, Vol. 2, No. 1, pp.65–104.

Burn, J.M. and Szeto, C. (2000) 'A comparison of the views of business and it management on success factors for strategic alignment', *Information & Management*, Vol. 37, No. 4, pp.197–216.

Cannon, J. and Byers, M. (2006) 'Compliance deconstructed', *ACM Queue*, Vol. 4, No. 7.

Chakraborty, D. and Lei, H. (2004) 'Pervasive enablement of business processes', *Proceedings of the International Conference on Pervasive Computing (PerCom)*, DC, USA, p.87.

Chan, Y.E., Sabherwal, R. and Thatcher, J.B. (2006) 'Antecedents and outcomes of strategic IS alignment: an empirical investigation', *IEEE Transactions on Engineering Management*, Vol. 53, No. 1, pp.27–47.

Cholewka, D.G., Botha, R.A. and Eloff, J.H.P. (2000) 'A context-sensitive access control model and prototype implementation', *Proceedings of the IFIP TC11 15th Annual Working Conference on Information Security for Global Information Infrastructures*, Vol. 47, pp.341–350.

Consolvo, S., Roessler, P., Shelton, B.E., LaMarca, A., Schilit, B. and Bly, S. (2004) 'Technology for care networks of elders', *IEEE Pervasive Computing*, Vol. 3, No. 2, pp 22–29.

Corradi, A., Montanari, R. and Tibaldi, D. (2004) 'Context-based access control management in ubiquitous environments', *Proceedings of the Third IEEE International Symposium on Network Computing and Applications (NCA)*, pp.253–260.

Cortadella, J., Kishinevsky, M., Lavagno, L. and Yakovlev, A. (1998) 'Deriving Petri nets from finite transition systems', *IEEE Transactions on Computers*, Vol. 47, No. 8, pp.859–882.

Cuppens, F. and Cuppens-Boulahia, N. (2008) 'Modeling contextual security policies', *International Journal of Information Security*, Vol. 7, No. 4, pp.285–305.

Damianides, M. (2004) 'How does SOX change IT?', *Journal of Corporate Accounting & Finance*, Vol. 15, No. 6, pp.35–41.

Ferraiolo, D.F., Kuhn, D.R. and Chandramouli, R. (2007) *Role-Based Access Control*, 2nd ed., Artech House, Norwood, MA.

Georgiadis, C.K., Mavridis, I., Pangalos, G. and Thomas, R.K. (2001) 'Flexible team-based access control using contexts', *Proceedings of the 6th ACM Symposium on Access Control Models and Technologies (SACMAT)*, pp.21–27.

Giner, P., Cetina, C., Fons, J. and Pelechano, V. (2011) 'Implicit interaction design for pervasive workflows', *Personal and Ubiquitous Computing*, Vol. 15, No. 4, pp.399–408.

Gruhn, V., Köhler, A. and Klawes, R. (2007) 'Modeling and analysis of mobile business processes', *Journal of Enterprise Information Management*, Vol. 20, No. 6, pp.657–676.

Hung, J.C., Wang, Y-B., Deng, L.Y., Yang, S. and Huang, C-H. (2010) 'A framework of online chain store integrated with personalised recommendation for e-commerce', *International Journal of Wireless and Mobile Computing*, Vol. 4, No. 3, pp.28–39.

Joshi, J.B.D., Bertino, E., Latif, U. and Ghafoor, A. (2005) 'A generalized temporal role-based access control model', *IEEE Transactions on Knowledge and Data Engineering*, Vol. 17, No. 1, pp.4–23.

Kapitsaki, G.M. and Venieris, I.S. (2008) 'PCP: Privacy-aware Context Profile towards context-aware application development', *Proceedings of the 10th International Conference on Information Integration and Web-based Applications & Services (iiWAS)*, pp.104–110.

Klarl, H., Marme, F., Wolff, C., Emig, C. and Abeck, S. (2009) 'An MDA-based environment for generating access control policies', *Trust, Privacy and Security in Digital Business*, Vol. 5695, pp.115–126.

Kulkarni, D. and Tripathi, A. (2008) 'Context-aware role-based access control in pervasive computing systems', *Proceedings of the 13th ACM Symposium on Access Control Models and Technologies (SACMAT)*, pp.133–122.

Leavitt, N. (2011) 'Mobile security: finally a serious problem?', *Computer*, Vol. 44, No. 6, pp.11–14.

Miller, C. (2011) 'Mobile attacks and defence', *IEEE Security and Privacy*, Vol. 9, No. 4, pp. 68–70.

Mishra, S. and Weistroffer, H. (2007) 'A framework for integrating Sarbanes-Oxley compliance into the systems development process', *Communications of the Association for Information Systems (CAIS)*, Vol. 20, No. 1.

Mouratidis, H. and Jürjens, J. (2010) 'From goal-driven security requirements engineering to secure design', *International Journal of Intelligent Systems*, Vol. 25, No. 8, pp.813–840.

Murata, T. (1989) 'Petri nets: properties, analysis and applications', *Proceedings of the IEEE*, pp.541–580.

Neubauer, T., Klemen, M. and Biffl, S. (2006) 'Secure business process management: a roadmap', *Proceeding of the 1st International Conference on Availability, Reliability and Security (ARES)*, pp.457–464.

Neumann, G. and Sobernig, S. (2009) 'Xotcl 2.0 – a ten-year retrospective and outlook', *Proceedings of the Sixteenth Annual Tcl/Tk Conference*, Portland, Oregon.

Neumann, G. and Sobernig, S. (2011) 'An overview of the next scripting toolkit', *Proceedings of the 18th Annual Tcl/Tk Conference*, Manassas, Virginia, USA.

Neumann, G. and Zdun, U. (2000) 'Xotcl, an object-oriented scripting language', *Proceedings of Tcl2k: the 7th USENIX Tcl/Tk Conference*, Austin, Texas, USA.

Neumann, G. and Zdun, U. (2012) *Xotcl Homepage*. Available online at: http://www.xotcl.org/

Oh, S. and Park, S. (2003) 'Task-role-based access control model', *Information Systems*, Vol. 28, No. 6, pp.533–562.

OMG (2010) *Object Constraint Language Specification*. Available online at: http://www.omg.org/spec/OCL/

OMG (2011a) 'Business Process Modeling and Notation (BPMN). Available online at: http://www.omg.org/spec/BPMN/

OMG (2011b) *Unified Modeling Language (OMG UML): Superstructure*. Available online at: http://www.omg.org/spec/UML/

Ousterhout, J. (1990) 'Tcl: an embeddable command language', *Proceedings of the 1990 Winter USENIX Conference*, January 1990.

Ouyang, C., Dumas, M., Aalst, W.M.P.V.D., Hofstede, A.H.M.T. and Mendling, J. (2009) 'From business process models to process-oriented software systems', *Transactions on Software Engineering and Methodology (TOSEM)*, Vol. 19, No. 1.

Rosemann, M., Recker, J.C. and Flender, C. (2008) 'Contextualisation of business processes', *International Journal of Business Process Integration and Management*, Vol. 3, No. 1, pp.47–60.

Roussos, G. (2006) 'Ubiquitous computing for electronic business', in Roussos, G. (ed.,): *Ubiquitous and Pervasive Commerce*, Computer Communications and Networks, Springer London, London, UK.

Sandhu, R., Coyne, E., Feinstein, H. and Youman, C. (1996) 'Role-based access control models', *IEEE Computer*, Vol. 29, No. 2, pp.38–47.

Sandhu, R.S. and Samarati, P. (1994) 'Access control: principles and practice', *IEEE Communications Magazine*, Vol. 32, No. 9, pp.40–48.

Schefer-Wenzl, S. and Strembeck, M. (2012) 'Modeling context-aware RBAC models for business processes in ubiquitous computing environments', *Proceedings of the 3rd International Conference on Mobile, Ubiquitous, and Intelligent Computing (MUSIC)*, *IEEE*, Vancouver, BC, pp.126–131.

Shabtai, A., Fledel, Y., Kanonov, U., Elovici, Y., Dolev, S. and Glezer, C. (2010) 'Google Android: a comprehensive security assessment', *IEEE Security and Privacy*, Vol. 8, No. 2, pp.35–44.

Sheng, Q.Z. and Benatallah, B. (2005) 'ContextUML: a UML-based modeling language for model-driven development of context-aware web services development', *Proceedings of the International Conference on Mobile Business*, pp.206–212.

Simons, C. (2007) 'CMP: a UML context modeling profile for mobile distributed systems', *Proceedings of the 40th Annual Hawaii International Conference on System Sciences (HICSS)*, p.289b.

Stewart, W., Xiao, Y., Sun, B. and Chen, H-H. (2007) 'Security mechanisms and vulnerabilities in the IEEE 802.15.3 wireless personal area networks', *International Journal of Wireless and Mobile Computing*, Vol. 2, No. 1, pp.14–27.

Strembeck, M. (2005) 'Embedding policy rules for software-based systems in a requirements context', *Proceedings of the 6th IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY)*, pp.235–238.

Strembeck, M. (2010) 'Scenario-driven role engineering', *IEEE Security & Privacy*, Vol. 8, No. 1, pp.28–35.

Strembeck, M. and Mendling, J. (2010) 'Generic algorithms for consistency checking of mutual-exclusion and binding constraints in a business process context', *Proceedings of the 18th International Conference on Cooperative Information Systems (CoopIS)*, Vol. 6426, Springer Verlag, pp.204–221.

Strembeck, M. and Mendling, J. (2011) 'Modeling process related RBAC models with extended UML activity models', *Information and Software Technology*, Vol. 53, No. 5, pp.456–483.

Strembeck, M. and Neumann, G. (2004) 'An integrated approach to engineer and enforce context constraints in RBAC environments', *ACM Transactions on Information and System Security (TISSEC)*, Vol. 7, No. 3, pp.392–427.

Tan, K., Crampton, J. and Gunter, C.A. (2004) 'The consistency of task-based authorization constraints in workflow systems', *Proceedings of the 17th IEEE Workshop on Computer Security Foundations*, pp.155–169.

Thomas, R.K. (1997) 'Team-based access control (TMAC): a primitive for applying role-based access controls in collaborative environments', *ACM Workshop on Role-Based Access Control*, pp.13–19.

Warner, J. and Atluri, V. (2006) 'Inter-instance authorization constraints for secure workflow management', *Proceedings of the 11th ACM Symposium on Access Control Models and Technologies (SACMAT)*, pp.190–199.

Wedde, H.F. and Lischka, M. (2004) 'Role-based access control in ambient and remote space', *Proceedings of the 9th ACM Symposium on Access Control Models and Technologies (SACMAT)*, pp.21–30.

Wolter, C. and Schaad, A. (2007) 'Modeling of task-based authorization constraints in BPMN', in Alonso, G., Dadam, P. and Rosemann, M. (Eds): *Business Process Management*, Vol. 4714, Springer Berlin/Heidelberg, pp.64–79.

Xiao, Y., Bandela, C., Du, X., Pan, Y. and Dass, E.K. (2006) 'Security mechanisms, attacks and security enhancements for the IEEE 802.11 WLANs', *International Journal of Wireless and Mobile Computing*, Vol. 1, No. 3, pp.276–288.

Zdun, U., Strembeck, M. and Neumann, G. (2007) 'Object-based and class-based composition of transitive mixins', *Information and Software Technology*, Vol. 49, No. 8, pp.871–891.

Zhang, X., Li, J. and Chen, Z. (2012) 'Behaviour adaptation of context aware web services in pervasive computing environment', *International Journal of Wireless and Mobile Computing*, Vol. 5, No. 2, pp.184–190.