Proteus
A Semantic Context-Aware Adaptive Policy Model

***Alessandra Toninelli*, Rebecca Montanari**
Department of Electronics, Computer Science and Systems (DEIS)
University of Bologna – Italy
{atoninelli, rmontanari}@deis.unibo.it

Ora Lassila
Nokia Research Center Cambridge
Cambridge, MA – USA
ora.lassila@nokia.com

Lalana Kagal
MIT CSAIL
Cambridge, MA – USA
lkagal@csail.mit.edu

IEEE Workshop on Policies for Distributed Systems and Networks – Bologna, Italy – June 14, 2007

Policy-Based Security & Behaviour Management in an Ad-Hoc Collaboration Scenario

The list of participants may not be known in advance and may change over time

Participants act as both resource providers and users

Participants may belong to different organizations

Participants are co-located (space and time)

The meeting may continue beyond the scheduled time, possibly in a different place

We need to integrate **context-awareness** into policies

- Definition of policies on the basis of context
- Adaptation of policies in response to context changes

Context-Awareness to Enable Policy Adaptation

Policy Adaptation

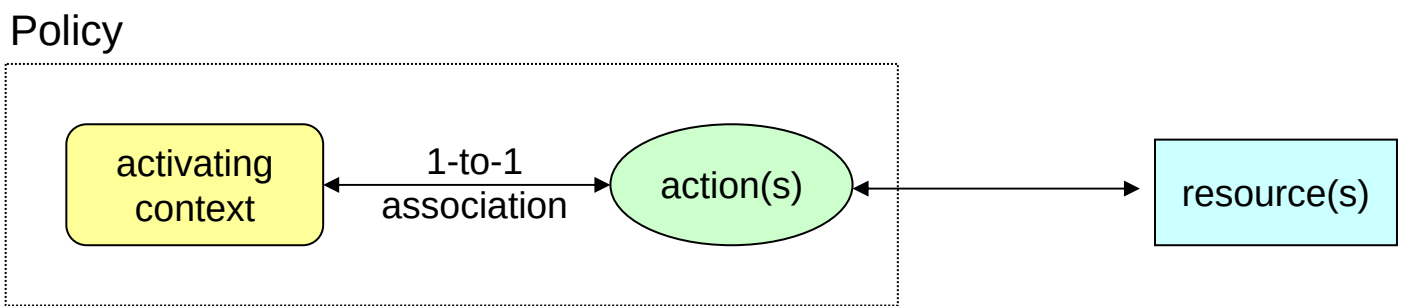
The ability for a policy-based security management system
to adjust policy specifications
in order to enable policy enforcement in different/unforeseen situations

- Requires appropriate modeling of situations (contexts) and policy elements
  - ↳ Context-aware policy model
- Requires reasoning over context and policy representations
  - ↳ Semantic technologies

Proteus Context-Aware Policy Model

- A context acts as intermediary between an entity and the set of actions that it can/must perform on resources.
- For each context, policies define permitted/obligated actions on resources.

  ⇨ **policy activating context**



- Entities can/must only perform those actions that are associated with *active* contexts

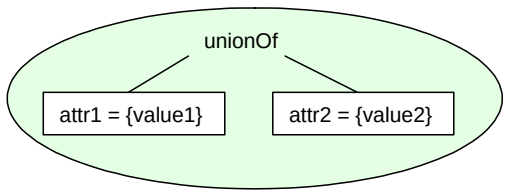
Context Activation Model

- An activating context is a set of attributes and predetermined values
  - single value or range of values
  - constant or variable values (wrt. context)
- The current state is a set of attribute/value pairs
  - snapshot values read from ”sensors”
- A context is *active* if

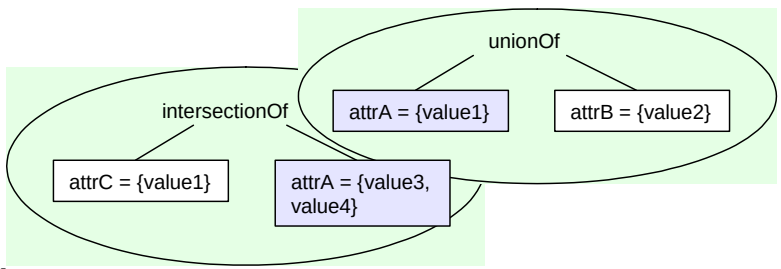
  the attribute values of the current state match the definition of the context

Semantic Context Model

- A *minimal context* consists of a single attribute/value assignment pair
- A *composite context* is the logical combination of several minimal contexts
  - unary relationships (is part of, negation)
  - n-ary relationships (conjunction, union)

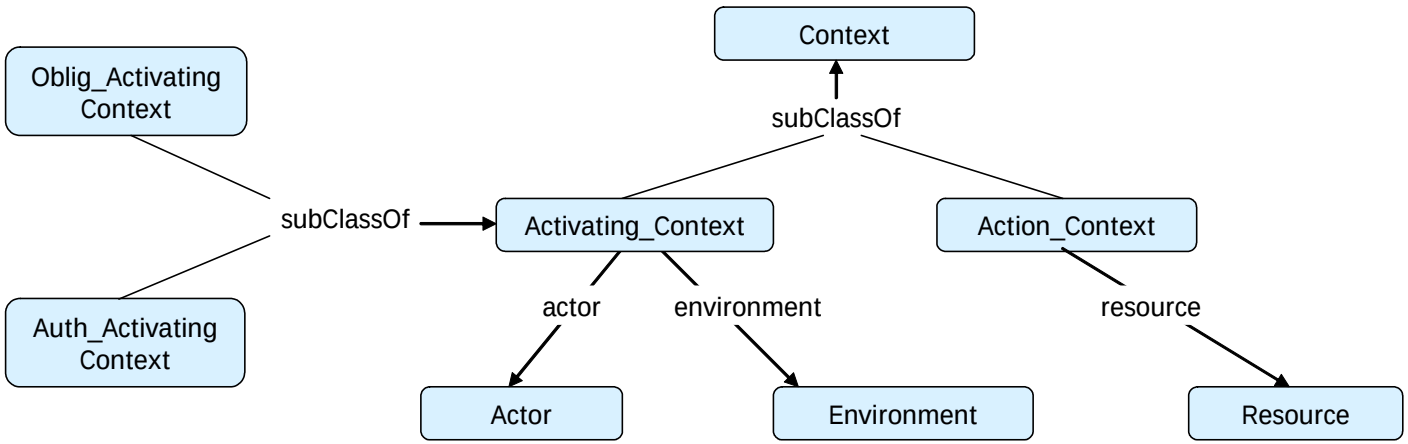


- Contexts might share one or more context attributes
  - *overlapping* contexts
  - *disjoint* contexts



- Contexts might or might not be *compatible*
  - attribute semantics
  - application-dependent constraints

DL Ontologies for Context Representation



- Authorization/obligation activating contexts are *classes*
  - OWL logical constructs: *subClassOf, intersectionOf, unionOf, disjointWith*
- Current state is an *instance* of context
- DL subsumption reasoning to determine active contexts

A Context-Aware Authorization Policy Example in a Spontaneous Collaboration Scenario

Access to the resources *related to the current* meeting is granted only *during* the meeting to any requestor that is *currently co-located* with the resource owner and is *currently working* on his set of meeting-*related* resources.

A Context-Aware Authorization Policy Example in a Spontaneous Collaboration Scenario

1. Context specification

Meeting Activating Context Specification
Meeting_Context ≡ Auth_Activating_Context ⊓ ∃owner.Meeting_Actor ⊓ ∃requestor.Co-located_Meeting_Actor
Meeting_Actor ≡ Actor ⊓ ∃is_currently_working_on.Current_Project ⊓ ∃located.Meeting_Space ⊓ ∃is_involved_in.Current_Project
Meeting_Action ≡ Access_Action ⊓ ∃action_context.Meeting_Action_Context
Meeting_Action_Context ≡ Action_Context ⊓ ∃resource.Current_Project_Resource

2. Policy specification

Meeting Policy Specification
Meeting_Policy ≡ Pos_Authorization_Policy ⊓ ∃controls.Meeting_Action ⊓ ∃activating_context.Meeting_Context

LP Rules for Context Representation

- DL-based reasoning may not be enough:
  1. We need to provide assertions to instantiate variable attribute values
  2. We need to correlate contexts via property path relationships between anonymous individuals
- We exploit Logic Programming-based reasoning by defining rules:

3. Rule specification

Colocated Meeting Actor Specification	
Colocated_Meeting_Actor ≡ ∃is_currently_working_on.Current_Project ⊓ ∃is_involved_in.Current_Project ⊓ ∃colocated_with.Resource_Owner	
Instantiation Rules to be applied in case of an ordinary scheduled meeting	
Current_Meeting_Rule	Scheduled_Calendar_Slot (?x) ∧ Meeting (?x) → Current_Meeting (?x)
Current_Project_Rule	Current_Meeting(?x) ∧ Project(?y) ∧ meeting_on_project(?x,?y) → Current_Project(?y)

Semantic Context-Aware Policy Adaptation

- Proteus supports three kinds of adaptation:
  - ⇨ Policy adaptation
  - ⇨ Action adaptation
  - ⇨ Context adaptation

- Policy adaptation consists in extending the validity of a policy in response to context changes

- Action adaptation consists in finding alternative actions in case an actor is not able to perform a permitted/obliged action

- Context adaptation consists in finding alternative contexts where permitted/obliged actions can be performed

Policy Adaptation Example

Colocated Meeting Actor Specification	
Colocated_Meeting_Actor ≡ ∃is_currently_working_on.Current_Project ⊓ ∃is_involved_in.Current_Project ⊓ ∃colocated_with.Resource_Owner	
Instantiation Rules to be applied in case of a meeting prolongation	
Current_Project_Rule-2	Actor(?y) ∧ Last_Current_Project(?x) ∧ is_currently_working_on(?y,?x) ∧ Scheduled_Calendar_Slot(?z) ∧ Idle(?z) → Current_Project(?x)
Current_Meeting_Rule-2	Scheduled_Calendar_Slot(?x) ∧ Idle(?x) ∧ Past_Calendar_Slot(?y) ∧ Meeting(?y) ∧ Current_Project(?z) ∧ meeting_on_project(?y,?z) → Current_Meeting(?y)

- What happens if the meeting goes beyond the scheduled time?

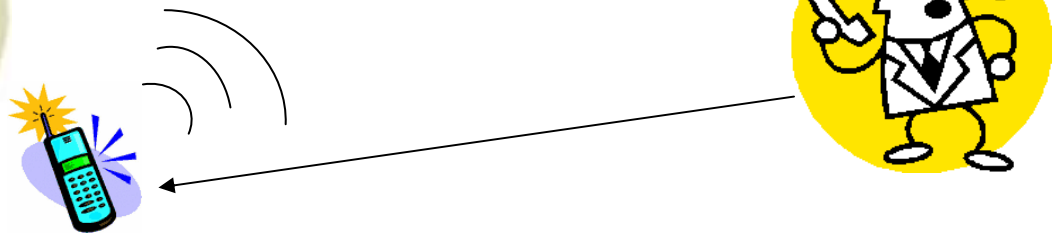
  ↳ By applying a different rule the same policy is still valid!

Action Adaptation

2. Her boss is trying to call her

1. Alice is at the meeting

4. Alice's phone is not able to send the SMS because of poor GSM network coverage.



3. Alice is not forwarded the call and she does not answer.

Whenever an incoming call from the boss during working time is not answered, an SMS must be sent to the boss.

When Alice is in meeting with a client, incoming calls are not authorized to be forwarded (e.g., by ringing).

- How to handle the (temporary) inability to perform the obliged action?

Action Adaptation Example

- Proteus allows to define semantically equivalent actions (wrt. context)
- DL + LP representation of the obligation policy
  - The obliged action has a variable value
  - The alternative values are instantiated by means of rules

Notification Policy Specification	
Boss_Notification_Policy ≡ Obligation_Policy ⊓ ∃triggers.Possible_Comm_Action_2Boss ⊓ ∃activating_context.No_Answered_Boss_Context	
Possible_Comm_Action_2Boss ≡ Possible_Communication_Action ⊓ ∃target.Boss	
Instantiation Rules to provide action adaptation	
Possible_Communication Rule-1	SendSMS_Action(?x) ∧ Actor(?y) ∧ is_able(?y,?x) → Possible_Communication_Action(?x)
Possible_Communication Rule-2	SendSMS_Action(?x)∧ Actor(?y) ∧ is_not_able(?y,?x) ∧ SendEmail_Action(?z) ∧ is_able(?y,?z) → Possible_Communication_Action(?z)

Context Adaptation

2. Her boss is trying to call her

1. Alice is at the meeting

4. Alice is not authorized to send the SMS.

Whenever an incoming call from the boss during working time is not answered, an SMS must be sent to the boss.

When Alice is in meeting with a client, incoming calls are not authorized to be

3. Alice is not forwarded the call and she does not answer.

When Alice has exhausted her corporate mobile credit, she is not authorized to make any call nor sending any SMS.

- An obliged action is not permitted
- Instead of changing the set of policies
  - ⇨ Proteus looks for a context that allows the obliged action
  - ⇨ and verifies its semantic relationship with the current context

Context Adaptation Example

(A 1+) Specification
A1_Policy ≡ Pos_Authorization_Policy ⊓ ∃controls.Call+SMS_Action ⊓ ∃activating_context.Valid_Credit_Context
(A 2-) Specification
A2_Policy ≡ Neg_Authorization_Policy ⊓ ∃controls.Call+SMS_Action ⊓ ∃activating_context.Not_Valid_Credit_Context
(A 3+) Specification
A3_Policy ≡ Pos_Authorization_Policy ⊓ ∃controls.Local_Call+SMS_Action ⊓ ∃activating_context.Xmas_Promotion_Context
Xmas_Promotion_Context ≡ Auth_Activating_Context ⊓ ∃environment.December_Env ⊓ ∃requestor.Promotion_Code_Employee

- The context of A3 cannot be activated
- The contexts of A2 is not compatible with the context of A1, but could be activated

Prototype Implementation

- The current prototype is implemented in Java
- Ontologies are specified in OWL-DL
- Rules are specified in SWRL
- The Pellet reasoner is accessed via OWL-API
  - to perform DL-based reasoning
  - to perform reasoning over DL-safe rules (Pellet SWRL support)

- The prototype has been tested in the collaborative scenario of a meeting
  - some performance issues with rule-based reasoning ☹
  - Pellet support to incremental reasoning just released (June 8, 2007)

Conclusions and Future Work

The Proteus policy model

- supports specification of context-aware policies
- allows policy adaptation in response to context changes

Future plans include:

- an implementation of the model using N3 and the cwm reasoner
- integration of techniques to identify and execute appropriate context transformation paths (e.g., planning techniques)
- exploring the role of trust for context-aware policy adaptation

- Example ontologies at http://www.lia.deis.unibo.it/research/Proteus/ontologies
- For the current Java prototype please contact atoninelli@deis.unibo.it